UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

TRABAJO FIN DE GRADO

Tagging privacy: cryptographic methods for avoiding statistical inference in personal data

Chen-Da Liu

Tutor: Simone Santini

MAYO 2014

"It is not knowledge, but the act of learning, not possession but the act of getting there, which grants the greatest enjoyment."

Carl Friedrich Gauss

Resumen

En los últimos años los avances en la tecnología y en Internet han resultado en un mundo en el que cualquier persona puede acceder a gran cantidad de información en la red. Un claro ejemplo se da con el uso de las redes sociales como Facebook o Twitter, que nos permiten encontrar gran cantidad de fotos etiquetadas de personas que no conocemos a priori.

En este trabajo nos situamos en el contexto en que una persona publica en la red una serie de datos (p.ej fotos) y metadatos (p.ej etiquetas de fotos). El uso de algoritmos avanzados de reconocimiento facial, junto con algoritmos estadísticos sencillos pueden dar a un intruso mucha más información de la que podríamos imaginar en una primera instancia. En concreto, se puede aprovechar la alta correlación de los datos y metadatos para mejorar los resultados de predicción en los algoritmos de reconocimiento facial, obteniendo así información precisa y concreta sobre la vida privada de un individuo.

Para evitar este tipo de ataques, proponemos un esquema en el que en primera instancia aleatorizamos los metadatos con el objetivo de disminuir la correlación existente, y después usamos un esquema de cifrado simétrico para dar seguridad sobre éstos.

Para medir dicha disminución en la correlación, usamos el Análisis de Correlación Canónica. Por otra parte, hemos analizado el comportamiento de la correlación tras aplicar los metadatos a distintas funciones, y diseñado una función a la que hemos llamado Función Cuadrática para eliminar la correlación de forma efectiva y eficiente.

Como la aleatorización de los metadatos no proporciona ningún tipo de confidencialidad usamos un esquema de cifrado simétrico convencional como el AES tras aleatorizar los metadatos. Sin embargo, estamos interesados en que determinados grupos de personas puedan acceder a la información original de determinados metadatos. Para ello, proponemos una generalización del protocolo de distribución de claves de Diffie-Hellman eficiente, que requiere un orden lineal del número de claves parciales transmitidas sobre el conjunto de personas que comparten la clave.

Dicha distribución de claves se propondrá en una topología convencional, en el que todos los miembros se pueden comunicar entre sí, y una topología centralizada, en donde todos los miembros se pueden comunicar únicamente con un distribuidor central. A su vez, hemos diseñado operaciones de adición y eliminación de miembros en los grupos, y hemos propuesto generalizaciones sobre los protocolos presentados.

PalabrasClave : Análisis de Correlación Canónica, Protocolo de Distribución de Claves, Diffie-Hellman, función de aleatorización, privacidad de etiquetado, correlación.

Abstract

In recent years, advances in technology and the Internet has resulted in a world where anyone can access a large amount of information on the net. A clear example is the use of social networks as Facebook or Twitter, which allow us to find lots of tagged photos of unknown people.

In this paper we place ourselves in a context in which someone publishes many pairs of data (ex. photos) and meta-data (ex. tags). Using advanced facial recognition algorithms, along with simple statistical algorithms a malicious intruder can get a lot more information than we could imagine at first. Specifically, one might take advantage of the high correlation between data and metadata to improve the prediction results in facial recognition algorithms, and thus obtaining precise and specific information about the life of an individual.

To prevent such attacks, we propose a scheme in which we randomize the metadata at first in order to decrease the correlation, and then we use a symmetric encryption scheme to provide security issues.

We use Canonical Correlation Analysis to measure the results of the decrease in the correlation. Moreover, we analyze how the correlation decreases after applying different randomization functions to meta-data. Also, we designed a specific function, which we denoted as Quadratic Function, to remove the correlation efficiently and effectively.

As randomization of the metadata does not provide any security, we use a symmetric encryption scheme such as AES after the process of randomization. We also want that friendly parties can get original information of certain metadata. In consecuence, we propose a generalization of the Diffie-Hellman key distribution protocol, which has a linear order of the number of partial keys transmitted over the set of people who share the key.

We propose this key distribution in the context of a conventional topology, in which all members can communicate with each other, or a centralized topology, in which all members can communicate only with a distributor. Finally, we designe operations to add and remove members of a group, and we propose some generalizations.

KeyWords : Canonical Correlation Analysis, Key agreement protocol, Diffie-Hellman, randomization function, tagging privacy, correlation.

Agradecimientos

El presente trabajo es un esfuerzo en el cual participaron varias personas dando ánimos y apoyo moral y psicológico.

Me gustaría que estas líneas sirvieran para expresar mi más profundo y sincero agradecimiento a todas aquellas personas que con su ayuda han colaborado en la realización del presente trabajo, en especial a Dr. Simone Santini, tutor de este proyecto, por la orientación, el seguimiento y la supervisión continua de la misma, pero sobre todo por la motivación y el apoyo recibido.

Un agradecimiento muy especial merece la comprensión, paciencia y el ánimo recibidos por parte de mi familia y amigos. En especial a mis padres Haoqing Liu y Xina Zhang, quienes me han apoyado en el largo proceso de mis estudios.

Contenidos

Lista de Figuras

Lista de Tablas

Acrónimos

DH	**D**iffie **H**ellman
CCA	**C**anonical **C**orrelation **A**nalysis
CCP	**C**oeficiente de **C**orrelación de **P**earson
PCA	**P**rincipal **C**omponent **A**nalysis
DLA	**D**iscrete **L**ogarithm **A**ssumption
CDH	**C**omputational **D**iffie **H**ellman
DDH	**D**ecision **D**iffie **H**ellman
CBA	**C**orrelation **B**ased **A**ttack

Símbolos

A, B	miembros concretos
X, Y, U, V	variables aleatorias
i, j	índices
m	mensaje en claro
$c(m)$	mensaje aleatorizado
S_i	sucesión i
a_n, b_n	términos n-ésimos de sucesiones
$E(X, Y)$	esperanza matemática de (X, Y)
$cov(X, Y)$	covarianza de (X, Y)
I	matriz identidad
N	número de bits
L	número de observaciones
G	grupo algebraico
n	número de participantes en distribución de claves
A	distribuidor de la información centralizada
M_i	i-ésimo miembro del grupo
g	generador de G
q	orden del grupo G
N_i	clave privada de M_i
N_a	clave privada de A
$\overline{N_i}$	N_i no incluido
K	clave generada
$\rho(x, y)$	correlación entre x e y
μ_x	media de x

σ_x	desviación típica de x
λ	autovalor
χ^2	test de hipótesis Chi Cuadrado
Λ	test estadístico Wilk's Lambda
α	generador de G
$\approx_{poly}$	indistinguibilidad computacional

Dedicado a mi familia, quienes me han apoyado para poder llegar a esta instancia de mis estudios.

Capítulo 1

En este trabajo

1.1 Contexto

Nos situamos en el contexto en que un individuo A al que llamaremos distribuidor tiene la intención de publicar un gran número de pares (x_i, y_i), a los que denotaremos por *instancias* de (X, Y), que son los *datos* y *metadatos* respectivamente.

Por ejemplo, supongamos que el individuo A esté publicando en internet fotos de sus amigos, sitios que visita, etc. A día de hoy es típico etiquetar dichas fotos con el nombre de esas personas o sitios. Supongamos que el distribuidor ha etiquetado a su amigo John Kennedy en varias fotos. Éste podría pensar que no ha publicado demasiada información sobre su amigo John debido a que se trata solamente de algunas fotos. Sin embargo, usando técnicas avanzadas de reconocimiento de caras y de minería de datos, un intruso hostil puede sacar mucha más información de lo que el distribuidor podría prever en primera instancia. Usando técnicas de reconocimiento facial se puede saber con qué personas estuvo John, y, usando la fecha de publicación de la foto el intruso podría saber cuándo e incluso dónde, dependiendo de la información disponible.

Usando técnicas estadísticas muy sencillas podemos conocer los lugares que John frecuenta de forma habitual y con quién [24]. Con esto podríamos saber mucho sobre la vida privada de John. Por otra parte, usando técnicas más sofisticadas de tomografía de red [25, 26] podríamos conocer estadísticamente los recorridos habituales de John y de sus compañeros, y saber aún más sobre su vida: con qué personas suele encontrarse y dónde tras salir del trabajo, a dónde va cuando está de vacaciones en Martha's Vineyard, etc.

En conclusión, las fotos publicadas por el distribuidor de forma inocente, pueden resultar en una desastrosa revelación de la vida privada de John.

El punto débil de estas consideraciones es el reconocimiento de la cara de John en las fotos. A pesar de los avances realizados, los algoritmos de reconocimiento facial actuales no proporcionan un reconocimiento fiable cuando tratamos con fotos que tienen cambios en la iluminación, mala orientación, o elementos influyentes (gafas, sombrero, barba, etc) [27]. Por tanto, los algoritmos estadísticos tendrán que trabajar en muchos casos con datos muy ruidosos, que limitarán su eficacia.

La situación cambia si añadimos metadatos fácilmente reconocibles como etiquetas.

En este ejemplo en concreto podemos pensar que miles de pares foto-etiqueta están disponibles, y nos interesamos por saber si un intruso puede obtener información de una persona concreta. Solamente mirando al contenido de las etiquetas, un intruso puede obtener por ejemplo, muchas imágenes de John Kennedy desde distintas perspectivas, que pueden ser almacenados en una base de datos asociada a esta persona. Además, las etiquetas, al no tener ruido por su naturaleza, nos proporcionan información fiable para aplicar los métodos estadísticos que ya hemos hablado.

Esto se podría extender a otras imágenes que incluso no estén etiquetadas. A pesar de que a día de hoy los algoritmos de identificación y reconocimiento biométrico del rostro de personas pueden no tener una tasa de aciertos muy alta, su eficacia aumenta si tenemos muchos ejemplos de la cara que queremos reconocer.

Por otra parte, si la correlación entre las fotos y las etiquetas es alta, un intruso podría aprovechar este hecho para sacar información visual a partir de las etiquetas.

Queremos evitar este tipo de situaciones. En particular, queremos evitar que un tercer individuo infiera información no autorizada usando técnicas estadísticas.

Dado un par de datos (x, y), llamamos ataque basado en correlación (CBA-attack) en x cualquier sistema que intente usar una estimación del valor y y la correlación entre y y x para estimar el valor de x.

A continuación presentamos un ejemplo para ilustrar el concepto de CBA, en el que x e y son números reales.

Vamos a asumir que los errores a la hora de medir x e y están distribuidos de forma gaussiana con desviaciones típicas σ_x y σ_y, con $\sigma_y \ll \sigma_x$.

Consideremos en primera instancia que medimos x directamente. Sin pérdida de generalidad asumimos que la medida real de x es 0. Entonces, tenemos:

$$p(x|0) = \frac{1}{\sqrt{2\pi\sigma_x^2}} e^{-\frac{x^2}{\sigma_x^2}}$$

O bien:

$$log(p(x|0)) = C\left(-\frac{x^2}{\sigma_x^2}\right)$$

donde C es una constante de normalización.

Por tanto, nuestra estimación es que $x = 0$, y el error cuadrático medio será σ_x^2.

Supongamos ahora que existe una correlación $\rho(x, y)$ entre x e y. Podemos medir y (con mejor precisión que x, ya que $\sigma_y \ll \sigma_x$) y usar la relación entre x e y para estimar el valor de x.

La fórmula de Bayes nos dice:

$$p(x|y) = \frac{p(x, y)}{p(y)}$$

y si suponemos de nuevo sin pérdida de generalidad que la medida de y es 0, tenemos:

$$p(x|0) = \frac{p(x|0, y|0)}{p(y|0)}$$

es decir,

$$log(p(x|0)) = log(p(x|0, y|0)) - log(p(y|0))$$

donde

$$p(y|0) = \frac{1}{\sqrt{2\pi\sigma_y^2}} e^{-\frac{y^2}{\sigma_y^2}}$$

$$p(x|0, y|0) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} e^{-\frac{1}{2(1-\rho^2)}\left(\frac{x^2}{\sigma_x^2} - \frac{2\rho xy}{\sigma_x\sigma_y} + \frac{y^2}{\sigma_y^2}\right)}$$

La correlación se puede relacionar con una función lineal entre x y y. Asumamos que esta dependencia es de la forma $x = \beta y$, y definimos:

$$\sigma = \beta\sigma_y$$

Tenemos:

$$log(p(x|0,y|0)) \propto -\frac{1}{2(1-\rho^2)}\left[\frac{1}{\sigma_x^2} - \frac{2\rho}{\beta\sigma_x\sigma_y} + \frac{1}{\beta^2\sigma_y^2}\right]x^2$$

$$= -\frac{1}{2(1-\rho^2)}\left[\frac{1}{\sigma_x^2} - \frac{2\rho}{\sigma_x\sigma} + \frac{1}{\sigma^2}\right]x^2$$

$$log(p(x|0,y|0)) - log(p(y|0)) \propto -\left[\frac{1}{2(1-\rho^2)\sigma_x^2} - \frac{\rho}{\sigma_x\sigma(1-\rho^2)} + \frac{1}{2(1-\rho^2)\sigma^2} - \frac{1}{\sigma^2}\right]x^2$$

$$= -\frac{1}{(1-\rho)^2}\left[\frac{1}{2\sigma_x^2} - \frac{\rho}{\sigma_x\sigma} + \frac{1/2-(1-\sigma)^2}{\sigma^2}\right]x^2 = -\frac{x^2}{\sigma_{eff}}$$

donde σ_{eff} es la desviación típica de la estimación de x a través de y.

Esta estimación mejora la estimación directa si $\sigma_{eff} < \sigma_x$, o bien, $\frac{1}{\sigma_{eff}} > \frac{1}{\sigma_x}$. Es decir:

$$(\frac{1}{\sigma_x^2} - \frac{1}{\sigma^2})\rho^2 + (\frac{2}{\sigma^2} - \frac{1}{\sigma_x\sigma})\rho - \frac{1}{2}(\frac{1}{\sigma_x^2} + \frac{1}{\sigma^2}) > 0$$

Esta es una ecuación de segundo grado que podemos resolver para ρ y obtener dos soluciones, una positiva y una negativa. Definiendo $v = \frac{\sigma}{\sigma_x}$, obtenemos una mejor estimación si:

$$\rho > \frac{1}{\sqrt{2}} + \frac{1}{2}\frac{v}{v^2+1}$$

Si estimamos y de forma exacta ($v = 0$), entonces mejoramos la estimación de x siempre y cuando $\rho > \frac{1}{\sqrt{2}}$. Además, a medida que aumenta σ, la correlación que necesitamos para obtener información útil es mayor.

Si queremos prevenir que un intruso malicioso quiera usar medidas de y para inferir información sobre x, tenemos dos posibilidades: incrementar la desviación típica de y o reducir la correlación ρ.

Incrementar la desviación típica es en general difícil: dado un dato y, el adversario usará los mejores algoritmos disponibles. Por tanto, a medida que se mejoran los algoritmos, se espera que σ_y disminuya. La alternativa es reducir la correlación ρ, que es de lo que nos ocuparemos en este trabajo.

Debemos observar que usar la Criptografía para cifrar directamente no necesariamente resuelve el problema. En nuestro ejemplo de John Kennedy, a pesar de que el cifrado

esconde la identidad de la persona representada en la foto, éste a priori no rompe la correlación entre las etiquetas y las fotos.

De esta manera, el intruso malicioso puede crear una base de datos grande de imágenes de esta persona, a pesar de que aún no sepa quién es. Si resulta que cualquier imagen de la misma persona se publica con el tag descifrado $'John\,Kennedy'$, la gran colección de imágenes de la base de datos se pueden usar para determinar (a través de algoritmos de reconocimiento facial basados en ejemplos) que la persona desconocida es, de hecho, John Kennedy rompiendo así la codificación. Observemos además que incluso aunque Kennedy nunca aparezca etiquetado, su identidad puede deducirse a partir de textos que contengan su nombre (e.g. artículos de periódicos, revistas, etc.)

Inspirados por este ejemplo, vamos a suponer que la distribución que sigue la aparición de John Kennedy sobre las fotos publicadas es $F_{xz} = P(X|Z)$, donde X es la variable aleatoria que indica si está John Kennedy en la foto, y Z la variable aleatoria que indica el conjunto de fotos publicadas. Por otra parte, consideramos $F_{yt} = P(Y|T)$ que es la función de distribución que indica que en la etiqueta aparece el nombre John Kennedy dadas las etiquetas publicadas. Podemos suponer que la varianza de la primera distribución es alta, ya que tratamos con fotos, y la segunda varianza es baja, o casi nula. Obsérvese que si la correlación entre las etiquetas y las fotos es alta, no importa que la varianza de la primera distribución sea baja, porque se podrían usar las etiquetas para sacar información de las fotos.

Por tanto, queremos anular la correlación que pudiera existir entre las fotos y las etiquetas. Por otra parte, queremos que determinados grupos amigables puedan acceder al contenido de las etiquetas.

La solución propuesta es la siguiente:

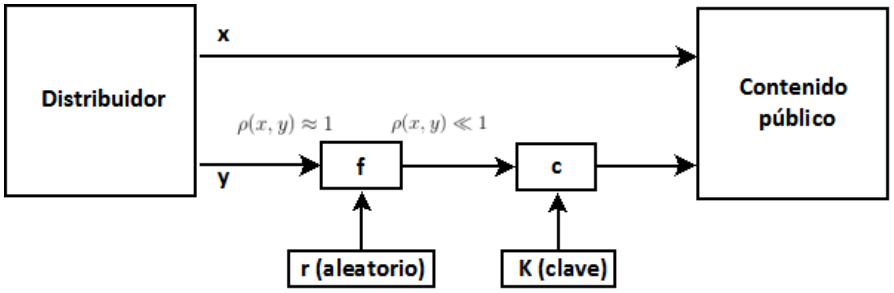

FIGURE 1.1: Esquema propuesto para resolver el problema

La función f es una función que aleatoriza las etiquetas, y provoca que la correlación entre las fotos y las etiquetas se pierda. Sin embargo, eso no resuelve el problema porque la función f podría ser reversible. Es por ello que usamos un esquema de cifrado

simétrico tras aplicar la función que aleatoriza, cuya clave distribuiremos proponiendo un protocolo extendido de Diffie-Helmann. Obsérvese que proponemos el uso de un criptosistema simétrico dada la naturaleza del problema: queremos que una persona codifique, y determinados grupos de personas decodifiquen. En un esquema tradicional de cifrado asimétrico, típicamente tenemos muchas personas codificando, y tan solo una decodificando.

Usamos un esquema de aleatorización previo al esquema de cifrado para poder controlar el comportamiento de la correlación y asegurar que éste disminuya de forma razonable.

Observamos que no realizamos ningún tipo de encriptación ni aleatorización al conjunto de datos x. Esto se debe a que en aplicaciones reales como las redes sociales, típicamente mostramos las imágenes en su forma original.

1.2 Estructura del documento

Para abordar este problema, hemos estructurado el documento en varios capítulos. En el primero de ellos presentaremos los algoritmos de aleatorización estudiados para ser usados en el criptosistema. Nos preocupamos por eliminar la posible correlación entre las etiquetas x, y las mismas etiquetas tras aplicar los algoritmos de aleatorización. La salida de dichos algoritmos se introducirá como entrada a un esquema de cifrado simétrico. Obsérvese que en realidad queremos disminuir la correlación entre las etiquetas originales y las imágenes, pero como la correlación entre éstos es muy alta, si tras aleatorizar las etiquetas éstas no tienen correlación con las etiquetas originales, tampoco tendrán correlación con las imágenes.

En el segundo capítulo nos preocupamos por comprobar que la correlación entre las etiquetas m y las etiquetas tras pasar el proceso de randomización $f(m)$ ha bajado. Para ello, estudiaremos el conocido Análisis de Correlación Canónica, una de las herramientas más generales que existen para estudiar correlaciones entre conjuntos, con hipótesis mínimas sobre dichos conjuntos.

En el tercer capítulo realizaremos una serie de pruebas que nos revelarán la efectividad de dichos algoritmos a la hora de ocultar la correlación.

En el cuarto capítulo, presentaremos una forma de distribuir claves para el conjunto de personas que quieran o necesiten acceder a la información de las etiquetas. Adicionalmente, definimos operaciones de adición y eliminación de miembros a dichos grupos.

Por último, mostraremos las conclusiones obtenidas y posibles trabajos futuros.

Capítulo 2

Algoritmos de aleatorización

2.1 Introducción

En esta sección nos preocupamos por saber cómo se comporta la correlación tras distintos procesos de aleatorización. Esto es, queremos crear una función que destruya la correlación de su entrada con su salida. En nuestro caso, la entrada serán las etiquetas de las fotos que vamos a publicar. De acuerdo al esquema propuesto en la figura 1.1, estamos diseñando la función f, para después introducir la salida en un esquema de cifrado. Recordemos que las funciones de aleatorización que vamos a presentar deben ser reversibles con facilidad, para poder recuperar la etiqueta original. Por otra parte, no queremos que el número de bits de salida sea mucho mayor que el número de bits en la entrada.

Vamos a presentar tres esquemas de aleatorización. El primero de ellos es un esquema que consiste en extender el número de bits en la salida, e introducir bits aleatorios en determinadas posiciones que fijamos al comienzo. El segundo esquema propuesto, al que denominamos Función Cuadrática, es una función que hemos diseñado de forma independiente con el objetivo de minimizar la correlación de forma eficiente. Finalmente, en el tercer esquema usamos el tradicional y conocido criptosistema One Time Pad con una clave generada de forma aleatoria.

2.2 Adición de bits en posiciones fijas

Supongamos que tenemos un mensaje en claro de n bits.

En la versión más sencilla, el mensaje aleatorizado tiene l bits, con $l > n$. De estos bits, n bits de posiciones fijas contendrán el mensaje original y los $l - n$ restantes contendrán

bits aleatorios. Para facilitar la comparación con los otros esquemas, hemos puesto $l = 2n$.

Método de randomización

Supongamos que el mensaje en claro es $m = (m_1, m_2, \cdots, m_n)$, y la clave es $k = (k_1, k_2, \cdots, k_{2n})$.

En la clave binaria hay n ceros y n unos.

Si denotamos por $\left\{k_{l_j} : j \in 1, \cdots, n\right\}$ el conjunto de unos de la clave k ordenados, es decir, $l_i < l_j$ si $i < j$, $\forall i, j$, entonces el mensaje aleatorizado es $f(m) = (f_1, f_2, \cdots, f_{2n})$, donde $\forall j \in 1, \cdots, 2n$:

$$f_j = \begin{cases} m_j \text{ si } k_{l_j} = 1 \\ a_j \text{ si } k_{l_j} = 0 \end{cases}$$

Donde a_j es un dígito binario elegido al azar con $P(0) = P(1) = \frac{1}{2}$.

Método de retorno

Para recuperar el mensaje original a partir del mensaje aleatorizado y la clave, basta con realizar una proyección del mensaje aleatorizado $c(m)$ sobre las componentes en los que clave contiene un uno. Es decir, si denotamos por $\left\{k_{l_j} : j \in 1, \cdots, n\right\}$ los unos de la clave k ordenados, es decir, $l_i < l_j \iff i < j, \forall i, j$ entonces el mensaje es:

$$m = (c_{l_1}, \cdots, c_{l_n})$$

2.3 Función cuadrática

En la función cuadrática que hemos diseñado, pretendemos romper las correlaciones que puedan encontrar con herramientas actuales como el Análisis de Correlación Canónica, cuyo procedimiento describiremos en el siguiente capítulo.

Para comprender la función, en primer lugar tenemos que describir la siguiente cadena:

$$S_0 = 0, 1, *, 3, *, *, 6, *, *, *, 10 \ldots$$

En dicha cadena, se empieza desde el número 0, y se van dejando i asteriscos cada vez que escribimos un número. Es decir, escribimos el 0 y dejamos cero asteriscos. Después el 1 y un asterisco, el 3 y dos asteriscos, etc.

El número que escribimos es el índice en el que se sitúa en la cadena contando asteriscos. Obsérvese que escribimos los asteriscos para una mejor visualización de la cadena, pero en realidad la cadena es una sucesión de números naturales sin asteriscos.

De esta forma, podemos definir una función $f : \mathbb{N} \to \mathbb{N}^2$, en donde a cada número natural, le asignamos infinitos números naturales de la siguiente forma:

- Al número 0 le asignamos los números de la cadena S_0.
- Al número 1 le asignamos los números naturales correspondientes al primer asterisco a la derecha de cada número de S_0. Si visualizamos el índice correspondiente al primer asterisco después de cada natural de la cadena S_0 (sin llegar al siguiente número natural de la cadena S_0), tenemos la cadena:

$$S_1 = *, *, 2, *, 4, *, *, 7, *, *, *, 11 \dots$$

- Al número n le asignamos el n-ésimo asterisco a la derecha de cada número de S_0 sin sobrepasar el siguiente número de la cadena.

Para aclarar esto, vamos a definir la función de forma explícita.

En primer lugar, queremos definir una fórmula general para la imagen de 0: $f(0) = S_0$.

Observamos que cada hueco entre dos números sucesivos va aumentando de uno en uno. Es decir, si escribimos $f(0) = S_0 = a_0, a_1, \cdots$, tenemos:

- $a_0 = 0$
- $a_1 = 1 = a_0 + 1$
- $a_2 = 3 = a_1 + 2$
- $a_3 = 6 = a_2 + 3$
- $\cdots$

Por tanto, se cumple la siguiente ecuación recurrente:

$$a_n = a_{n-1} + n,\ a_0 = 0$$

Es fácil observar a partir de aquí que:

$$a_n = a_{n-1} + n = a_{n-2} + (n-1) + n = \cdots = 1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}$$

Es decir, el n-ésimo término de la sucesión S_0 viene dado por $a_n = \frac{n(n+1)}{2}$.

A partir de la fórmula general del n-ésimo término de S_0, podemos calcular la fórmula de los términos de la imagen $f(i), \forall i$ simplemente sumando i en la fórmula general. Sin embargo, debemos observar que la sucesión $f(i)$ tiene su primer término a continuación del término i de S_0. Esto se debe a que los términos anteriores no tienen i asteriscos a la derecha previos al siguiente término de S_0.

En conclusión, la fórmula del término n-ésimo de la sucesión $S_i = f(i)$, que denotamos por b_{in} viene de sumar el término $(n+i)$-ésimo de S_0 con i. Es decir, $b_{in} = \frac{n(n+1)}{2} + i$, definido para $n \geq i$.

Si lo definimos para $n \geq 0$, basta con realizar una traslación, llegando a la fórmula:

$$b_{in} = \frac{(i+n)(i+n+1)}{2} + i, \ n \geq 0$$

A continuación escribimos algunas observaciones y propiedades de dicha codificación:

- Para poder definir la imagen $f(n)$, esto es, que la sucesión S_n tenga como mínimo un término, necesitamos por lo menos ser capaces de codificar el número $b_{n0} = \frac{n(n+1)}{2} + n$.
- Si queremos que la imagen de $f(n)$ tenga por lo menos k términos, necesitamos por lo menos $b_{nk} = \frac{(n+k)(n+k+1)}{2} + n$ números.
- Si usamos una codificación de n bits, podremos codificar hasta el número $2^n - 1$. Si escribimos $b_{0m} = \frac{m(m+1)}{2} \leq 2^n - 1$, $M = max_m \left\{ \frac{m(m+1)}{2} \leq 2^n - 1 \right\}$ será la cantidad de números naturales que podemos codificar con n bits.
- Si $f(n)$ tiene tiene 1 término, entonces $f(i)$ tiene $n - i + 1$, $0 \leq i \leq n$ términos.

Método de randomización

Supongamos que estamos operando con bloques en la salida de n bits, y que queremos codificar un mensaje en claro interpretado como número natural m menor o igual a M. Generamos una clave k entre 0 y K, donde K viene definida como:

$$K = max_i \left\{ \frac{(i+m)(i+m+1)}{2} + m \leq 2^n - 1 \right\}$$

A continuación codificamos m como:

$$b_{mk} = \frac{(m+k)(m+k+1)}{2} + m$$

Método de retorno

Como cada número de la imagen de f corresponde únicamente a un elemento del dominio, podemos calcular la inversa de la función.

Tenemos el número b y deseamos conocer los índices m y k tales que $b = b_{mk}$. Para ello, basta seguir los siguientes pasos:

1. Definimos $k^* = max \left\{ j \ : \ \frac{j(j+1)}{2} < b \right\}$.
2. El índice m es $b - \frac{k^*(k^*+1)}{2}$.
3. El índice k es $k^* - m$.

2.4 One Time Pad

Este famoso esquema fue diseñado originalmente como esquema de cifrado con la propiedad de ser secreto perfecto. Esto quiere decir desde la teoría de la información de Shannon que los textos planos y los textos cifrados son independientes. En este apartado vamos a usar este esquema de cifrado no para cifrar, sino para estudiar la pérdida de correlación.

En dicho algoritmo, se crea una clave de la misma longitud que el mensaje, de n bits, y se calcula el mensaje cifrado como la operación XOR del mensaje original y la clave.

Método de randomización

Supongamos que el mensaje en claro es $m = (m_1, m_2, \cdots, m_n)$, y la clave es $k = (k_1, k_2, \cdots, k_n)$, entonces el mensaje cifrado es:

$$f(m) = (m_1 \, XOR \, k_1, m_2 \, XOR \, k_2, \cdots, m_n \, XOR \, k_n)$$

Método de retorno

Dado que realizar la operación XOR dos veces con un mismo elemento es como no realizar ninguna operación, basta con reproducir el método de cifrado con la misma clave, y a partir del texto cifrado:

$$f(m)\,XOR\,k = (m_1\,XOR\,k_1\,XOR\,k_1, \cdots, m_n\,XOR\,k_n\,XOR\,k_n) = m$$

Capítulo 3

Análisis de la correlación canónica (CCA)

3.1 Introducción

En este bloque estudiamos una herramienta potente para detectar correlaciones entre variables aleatorias. El objetivo es usar esta herramienta concreta para buscar correlaciones entre los textos y los textos aleatorizados. De esta manera, en el siguiente capítulo usaremos esta herramienta para observar cómo controlamos la disminución de la correlación de los textos aleatorizados al pasar por la función f. Usaremos como funciones de aleatorización las distintas funciones propuestas en el Capítulo 2.

Hoy en día existen numerosas técnicas para estudiar la correlación entre variables aleatorias. Uno de los índices de correlación más famosos es el tradicional Coeficiente de Correlación de Pearson (CCP), una medida de la relación lineal entre dos variables aleatorias cuantitativas.

Dadas dos variables aleatorias X, Y, el CCP se define como:

$$\rho_(x, y) = \frac{cov(X, Y)}{\sigma_x \sigma_y} = \frac{E(X - \mu_x, Y - \mu_y)}{\sigma_x \sigma_y}$$

Donde $cov(X, Y)$ es la covarianza de (X, Y), σ_x la desviación típica de X, σ_y la desviación típica de Y.

Sin embargo, este coeficiente sólo nos permite realizar estudios lineales cuando tratamos con dos variables.

Cuando tratamos con órdenes de correlación superiores, es necesario estudiar asociaciones entre vectores multivariantes. Uno de los análisis que se ha puesto de moda en los últimos años, y que es considerado uno de los análisis de correlación más generales es el Análisis de Correlación Canónica (CCA).

A diferencia de otros métodos para el estudio de correlaciones como el CCP (compara una variable con otra) o Análisis de Regresión Múltiple (compara una variable con un vector de variables), CCA no necesita realizar suposiciones iniciales sobre el número de variables aleatorias en cada vector. Es decir, se usa típicamente para buscar la correlación entre dos conjuntos de variables.

Otros análisis como el Análisis de la Componente Principal (PCA) se ocupan de solamente un conjunto de variables.

En las siguientes secciones explicaremos el funcionamiento de la CCA de forma intuitiva y con formalizaciones matemáticas.

3.2 Funcionamiento de la CCA

El Análisis de Correlación Canónica parte de dos vectores $X = (X_1, X_2, \cdots, X_n)$ e $Y = (Y_1, Y_2, \cdots, Y_m)$, $n \leq m$ de variables aleatorias que no tienen por qué ser de la misma dimensión.

En primer lugar, CCA crea combinaciones lineales de las componentes de los vectores.

Definimos $U = (U_1, U_2, \cdots, U_n)$, $V = (V_1, V_2, \cdots, V_n)$, donde:

$$U_i = \sum_{j=1}^{n} a_{ij} X_j, 0 < i \leq n$$
$$V_i = \sum_{j=1}^{m} b_{ij} Y_j, 0 < i \leq n$$

Definimos como el i-ésimo par canónico el par (U_i, V_i). Nuestra intención es buscar los pesos a, b tales que maximicen las correlaciones entre los componentes de cada par canónico. Es decir, queremos maximizar:

$$\rho_i = \frac{cov(U_i, V_i)}{\sigma_{U_i} \sigma_{V_i}}$$

Vamos a proceder de esta manera:

Para el primer par (U_1, V_1), buscamos los coeficientes $a_{11}, a_{12}, \cdots, a_{1n}$ y $b_{11}, b_{12}, \cdots, b_{1m}$ tales que maximicen la correlación ρ_1, pero sujeto a que la varianza de cada variable es uno:

$$var(U_1) = var(V_1) = 1$$

Esto se realiza para que los pesos se elijan de manera única.

Para el segundo par (U_2, V_2), buscamos de nuevo los coeficientes $a_{21}, a_{22}, \cdots, a_{2n}$ y $b_{21}, b_{22}, \cdots, b_{2m}$ tales que maximizan la correlación ρ_2, pero esta vez tenemos además las condiciones de que las correlaciones entre las variables (U_i, V_j) sean 0. Es decir:

$$\begin{aligned} var(U_1) = var(V_1) = 1 \\ cov(U_1, U_2) = cov(V_1, V_2) = 0 \\ cov(U_1, V_2) = cov(V_1, U_2) = 0 \end{aligned}$$

En general, para el i-ésimo par canónico buscaríamos los coeficientes $a_{i1}, a_{i2}, \cdots, a_{in}$ y $b_{i1}, b_{i2}, \cdots, b_{im}$ tales que maximizan la correlación ρ_i. Las restricciones serían:

$$\begin{aligned} var(U_i) = var(V_i) = 1 \\ cov(U_i, U_j) = cov(V_i, V_j) = cov(U_i, V_i) = 0, \forall i, j, i \neq j \end{aligned}$$

3.3 Formalización matemática

Ahora que hemos explicado el Análisis de Correlación Canónica en términos conceptuales, procedemos a usar técnicas matemáticas para lograr el resultado deseado.

3.3.1 Matriz de correlaciones: Obtención de las raíces canónicas

Recordemos que CCA identifica patrones de correlación entre dos vectores multivariantes. En concreto, el problema consiste en encontrar una combinación lineal de n variables y una combinación lineal de m variables, de tal forma que la correlación total se maximice.

Esta correlación dependen de la naturaleza y forma tanto de las correlaciones existentes en un mismo conjunto, como de las correlaciones entre ambos conjuntos.

Estas correlaciones se pueden expresar en una única matriz:

$$R = \begin{pmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{pmatrix}$$

Donde:

- R_{11} es la matriz de dimensión $n \times n$ de correlaciones entre las variables del primer conjunto.
- R_{12} es la matriz $n \times m$ de correlaciones entre n las variables del primer conjunto y las m variables del segundo conjunto.
- R_{21} es la traspuesta de R_{12}.
- R_{22} es la matriz $m \times m$ de correlaciones entre las m variables del segundo conjunto.

Dicha matriz R expresa de forma básica los patrones de asociación existentes en el propio conjunto y entre ambos conjuntos.

Para encontrar las correlaciones, tenemos que resolver un problema de autovalores. Es decir, buscar las raíces λ de la siguiente ecuación:

$$det(M - \lambda I) = det(R_{22}^{-1} R_{21} R_{11}^{-1} R_{12} - \lambda I) = 0$$

Donde I es la matriz identidad, y $M = R_{22}^{-1} R_{21} R_{11}^{-1} R_{12}$.

Podemos pensar que en la matriz M se combinan las correlaciones de la matriz de correlaciones R.

Esta matriz M expresa el nivel de solapamiento que existe en los datos. Para medir si el solapamiento es grande o pequeño, se realiza el cálculo de sus autovalores.

Tras calcular las raíces canónicas de la ecuación $det(M - \lambda I) = 0$, queremos saber de qué forma se correlacionan los dos conjuntos de datos. Esto se interpreta de la siguiente manera:

- Si hay una raíz distinta de cero, ésta indica que un conjunto de variables pueden proyectarse sobre el otro conjunto de variables en una línea, dados los pesos apropiados.
- Si existen dos raíces, quiere decir que cada conjunto puede proyectarse sobre un plano del otro conjunto.

- De la misma forma, si existen n raíces los conjuntos se pueden proyectar sobre un hiperplano de dimensión n del otro conjunto.

3.3.2 Interpretación de las raíces canónicas

Tras completar los pasos anteriores nos interesamos por saber interpretar de alguna forma los resultados obtenidos.

En concreto vamos a examinar cada raíz extraída de la matriz M para saber cuánto de significativos son.

Para ello, emplearemos un test de hipótesis. En este contexto se emplea típicamente el test de hipótesis χ^2 de Bartlett.

Vamos a considerar que la hipótesis nula es que los dos conjuntos de datos son independientes, es decir, que no tienen ninguna correlación. Y como hipótesis alternativa, consideramos que los datos tienen correlación.

Sin embargo, el test χ^2 no se aplicará directamente a los datos. Éste se aplicará sobre el resultado del conocido test estadístico Wilk's Lambda (Λ).

El resultado de este test se puede obtener con las raíces que hemos calculado anteriormente con la siguiente fórmula:

$$\Lambda = \prod_{i=1}^{k}(1 - \lambda_i)$$

Es decir, la hipótesis nula es testeada por la función Wilk's Lambda, cuya distribución se puede aproximar por una χ^2 con nm grados de libertad:

$$\chi^2 = -\left[(N-1) - 0.5(p+q+1)\right] ln\Lambda$$

Si la hipótesis nula se puede rechazar, entonces los dos conjuntos de datos están correlacionados de forma significativa.

De esta forma, podemos eliminar la primera raíz, y comprobar de nuevo si la correlación se mantiene con las raíces restantes.

Este proceso se puede realizar para comprobar la importancia que tiene cada raíz.

3.3.3 Obtención de los vectores canónicos

Tras saber la influencia estadística de cada una de las raíces canónicas, podemos proceder a definir de forma empírica aquellos patrones más significativos.

La influencia de cada patrón X_i e Y_i, $0 < i \leq n$, se transmite a través de los pesos a_i, b_i, $a_i \in \mathbb{R}^n, b_i \in \mathbb{R}^m$ que se asignan en las combinaciones lineales para calcular los vectores canónicos asociados U_i, V_i.

El cálculo de los pesos b_i de las variables del segundo conjunto se realiza resolviendo la siguiente ecuación:

$$(M - \lambda_i I)b_i = 0$$

Y los pesos a_i se obtienen realizando la siguiente operación:

$$a_i = \frac{R_{11}^{-1} R_{12} b_i}{\lambda_i}$$

Una vez obtenidos los pesos, si queremos obtener los vectores canónicos U_i, V_i, simplemente tenemos que realizar las siguientes operaciones:

$$U_i = a_i \cdot X$$
$$V_i = b_i \cdot Y$$

Donde $\cdot$ denota el símbolo de producto escalar.

Capítulo 4

Pruebas CCA

4.1 Introducción

Para cada uno de los algoritmos descritos en el Capítulo 2, se realizan pruebas de correlación usando la técnica de Análisis de Correlación Canónica descrita en el capítulo anterior.

El planteamiento de las pruebas se realiza buscando correlaciones entre dos conjuntos: los textos en claro y textos aleatorizados.

Para ello, vamos a realizar estudios sobre las correlaciones que existen entre los bits de ambos conjuntos.

El primer conjunto de variables, $X = (X_1, \cdots, X_N)$, representa los mensajes sin aleatorizar como mensajes de bits. Es decir, N es el número de bits de los mensajes en claro. Por otra parte, el segundo conjunto de variables $Y = (Y_1, \cdots, Y_M)$ es el mensaje aleatorizado, que tendrá M bits.

Recordemos que el objetivo de CCA es maximizar la correlación entre cada par de vectores canónico (U_i, V_i), $0 < i \leq n$, donde $n = min\{rg(X), rg(Y)\}$.

En concreto, se supondrá que el mensaje en claro es de N bits (se corresponde con el número de componentes de X), y el número de observaciones o muestras es L. Este número de muestras es el mismo para X y para Y.

Se realizarán gráficas representando valores representativos sobre el comportamiento de la correlación.

En general representaremos el comportamiento de la correlación entre determinados pares de vectores canónicos, su media y su desviación típica, tanto variando el número

de bits en el texto en claro N, como variando el número de observaciones L. Además, observaremos cómo se comporta la correlación a la hora de establecer las restricciones de ortogonalidad que establece el Análisis de Correlación Canónica.

Obsérvese que las gráficas mostradas muestran de forma clara el comportamiento de la correlación, por lo que no ha sido necesario usar herramientas como el test χ^2 o el test estadístico Wilk's Lambda.

4.2 Adición de bits

Para este algoritmo generamos mensajes aleatorios de N bits, donde N varía entre 64 y 1024 bits, y un número de observaciones L que oscila entre 5 y 10.000.

Tras realizar las pruebas, se obtiene una correlación perfecta entre los mensajes en claro y los mensajes aleatorizados para cualquier número de bits y observaciones.

La correlación entre los vectores (U_i, V_i) es siempre 1.

A continuación mostramos una gráfica como ejemplo donde $N = 512$, $L = 1000$. Con el objetivo de visualizar con claridad la gráfica solamente se representan los pares de vectores canónicos (U_i, V_i), $i = \{5, 300, 500\}$:

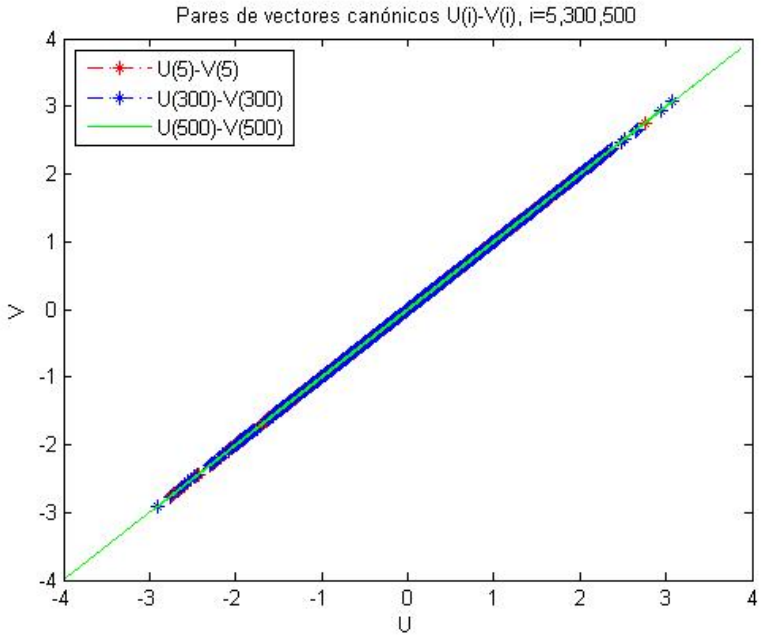

FIGURE 4.1: (U_i, V_i), $i = \{5, 300, 500\}$.

Observemos que X es una matriz 1000×512, e Y es una matriz 1000×1024, por lo que hay 512 pares canónicos (U_i, V_i). Es decir, las matrices U, V son matrices 1000×512.

La correlación no disminuye tras el algoritmo de aleatorización porque el Análisis de Correlación Canónica es capaz de proyectar ambos mensajes en un espacio en el que la correlación se mantiene. En concreto, si nos fijamos en la naturaleza del análisis, vemos que en el texto cifrado aquellos bits que corresponden al mensaje tendrán pesos perfectos, y los bits aleatorios tienen pesos nulos a la hora de hacer la combinación lineal.

Es por ello que el CCA es capaz de encontrar la correlación.

4.3 Algoritmo cuadrático

En este caso vamos a generar mensajes aleatorios de N bits, y vamos a reservar $2N$ bits para el texto cifrado. Obsérvese que esto nos permitirá definir sin problemas la imagen del número $2^N - 1$.

Queremos saber el comportamiento de la correlación cuando varía el número de bits N manteniendo el número de observaciones L fijo.

Para entender el comportamiento general de las correlaciones, hemos escogido presentar para cada N las correlaciones $r_1, r_N, r_{\frac{N}{2}}$ entre el primer par canónico (U_1, V_1), el último par canónico (U_N, V_N) y el par canónico intermedio $(U_{\frac{N}{2}}, V_{\frac{N}{2}})$. Por otra parte, hemos calculado la media y la desviación típica de las correlaciones.

A continuación mostramos qué sucede con los valores calculados en un ejemplo donde mantenemos el número de observaciones $L = 5000$, y variamos N entre los valores $N = 32, 64, 128, 256, 512$.

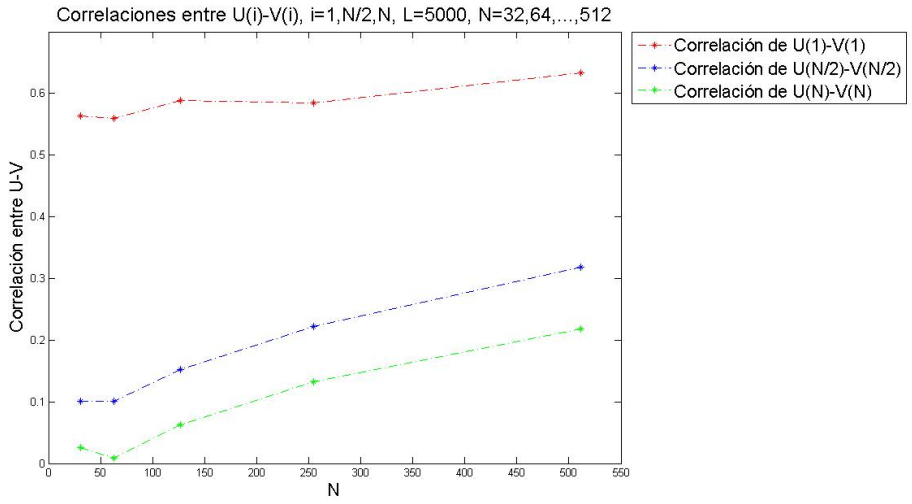

FIGURE 4.2: Correlación de pares canónicos primero, medio y último: $L = 5000$, $N = \{32, 64, 128, 256, 512\}$.

Observamos que las correlaciones aumentan ligeramente cuando el número de bits aumenta. Además, la correlación intermedia y la última se mantienen bajas. En concreto, la primera correlación se mantiene intermedia, rondando un 0.6, mientras que las otras correlaciones crecen en el intervalo $0 - 0.3$.

A continuación presentamos una gráfica donde se muestra cómo varía la media y la desviación típica:

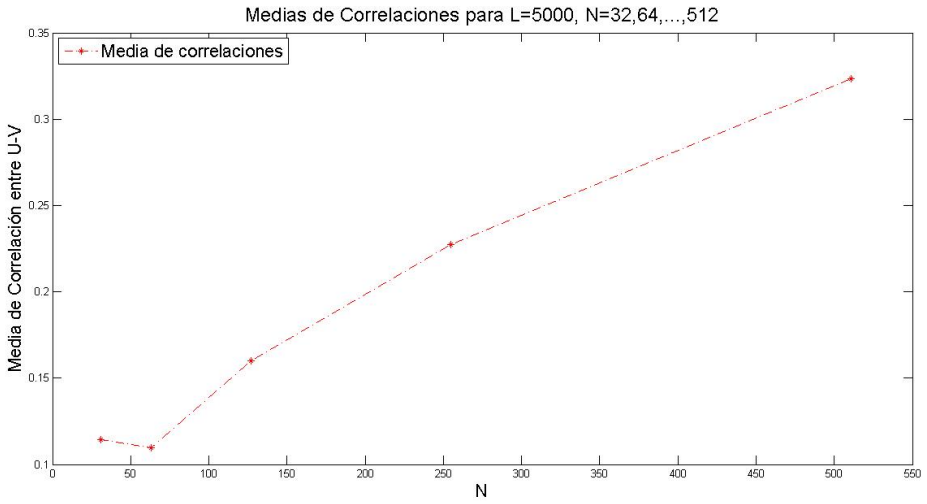

FIGURE 4.3: Media de correlaciones en los pares canónicos: $L = 5000$, $N = \{32, 64, 128, 256, 512\}$.

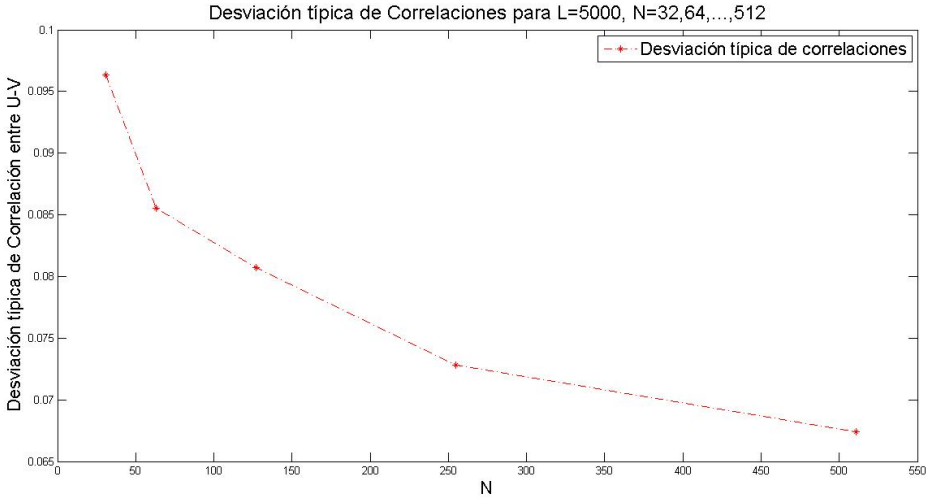

FIGURE 4.4: Desviación típica de correlaciones en los pares canónicos: $L = 5000$, $N = \{32, 64, 128, 256, 512\}$.

Observamos que la media de las correlaciones crece cuando N aumenta. Sin embargo, no logra superar un 0.35. La media de las correlaciones aumenta, porque si mantenemos

el número de observaciones fijo, la naturaleza de la Función Cuadrática se ve menos pronunciada si aumentamos el número de bits.

En cambio, la desviación típica decrece de forma exponencial, aunque se mantiene baja todo el tiempo (es siempre menor que 0.1).

En la segunda prueba observamos qué sucede cuando dejamos fijo el número de bits N, y variamos el número de observaciones L.

De nuevo, vamos a representar el comportamiento de las correlaciones $r_1, r_N, r_{\frac{N}{2}}$ que se dan en el primer par canónico (U_1, V_1), el último par canónico (U_N, V_N) y el par canónico intermedio $(U_{\frac{N}{2}}, V_{\frac{N}{2}})$. Además, hemos calculado la media y desviación típica de las respectivas correlaciones.

Para poder representar el comportamiento, hemos elegido como número de bits $N = 128$, y el número de observaciones variando entre $L = 1000 - 10000$.

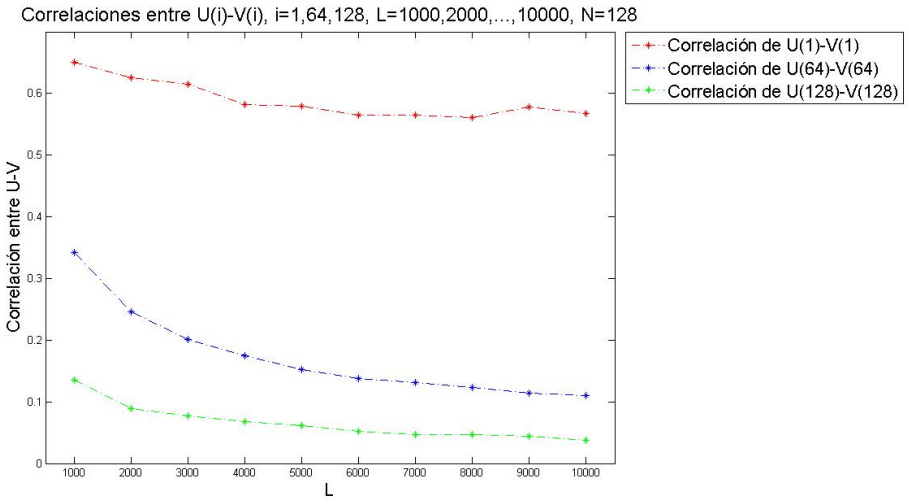

FIGURE 4.5: Correlación de pares canónicos primero, medio y último: $L = \{1000i, i = \{1 \cdots 10\}\}$, $N = 128$.

Observamos que esta vez las correlaciones de los pares canónicos primero e intermedio decrecen ligeramente en un rango variable.

La primera correlación ronda el valor del 0.6, mientras que las otras dos correlaciones convergen hacia valores del 0.1 la correlación del par intermedio, y 0.05 la del último par canónico.

A continuación, al igual que en la prueba anterior realizamos un estudio sobre la media y la desviación típica de la correlación a medida que aumenta el número de observaciones:

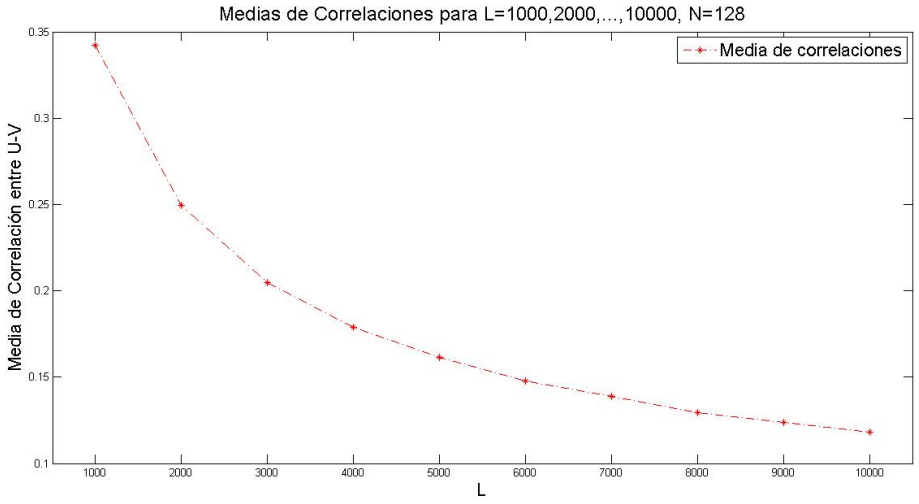

FIGURE 4.6: Media de correlaciones en los pares canónicos: $L = \{1000i, i = \{1 \cdots 10\}\}$, $N = 128$.

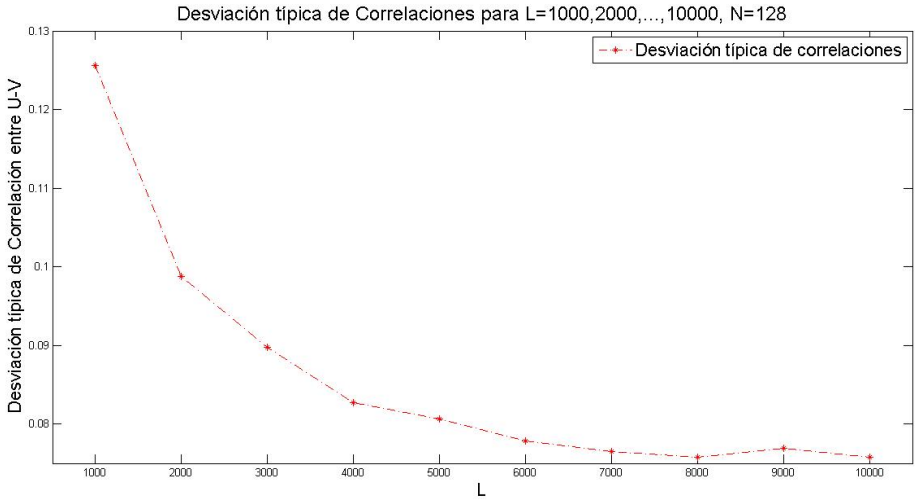

FIGURE 4.7: Desviación típica de correlaciones en los pares canónicos: $L = \{1000i, i = \{1 \cdots 10\}\}$, $N = 128$.

El valor de la media de las correlaciones baja desde 0.35 de forma exponencial hasta ser casi nula.

La desviación típica baja desde 0.12 hasta el rango $[0.075 - 0.8]$, donde se mantiene estable.

En la última prueba, nos interesamos por saber lo restrictivo que es el Análisis de Correlación Canónica al establecer las condiciones de que perpendicularidad en cada paso.

Para ello, representamos cómo se comporta la correlación a lo largo de los pares canónicos, para $N = 128$, $L = 1000$:

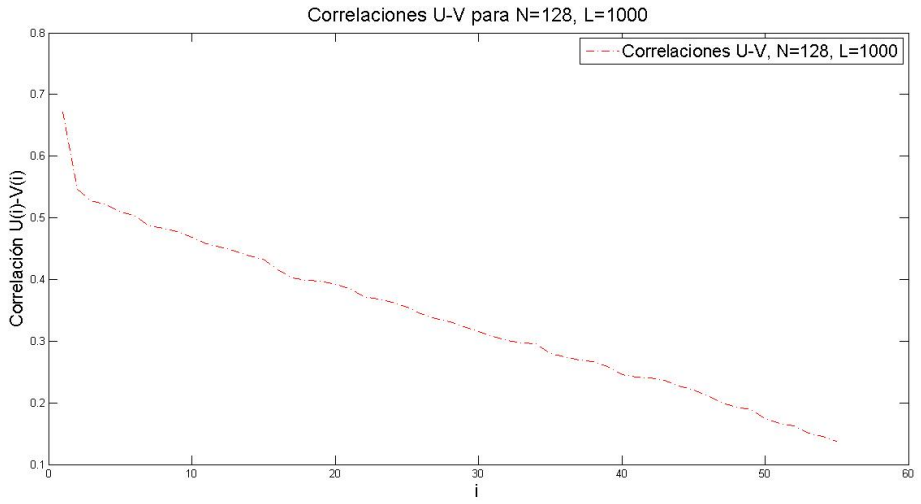

FIGURE 4.8: Correlaciones entre $U(i) - V(i)$: $i = \{1 \cdots 128\}$, $N = 128$, $L = 1000$.

Vemos que la correlación del primer par canónico tiene un valor casi de 0.7, pero al establecer las condiciones de perpendicularidad ésta la baja de hasta valores inferiores a 0.2.

La razón principal por la que el Análisis de Correlación Canónica no logra encontrar correlaciones en esta función, es que la función tiene una naturaleza no lineal. Por otra parte, la imagen de un mismo valor de entrada generalmente devuelve diferentes valores de salida.

Finalmente, hay que destacar que el número exacto de bits requeridos en la salida depende de la cantidad de imágenes que queremos que haya. Sin embargo, un sencillo cálculo nos muestra que con $2N$ bits podremos definir sin problemas todos los números de N bits.

4.4 One Time Pad

De nuevo generamos mensajes aleatorios de N bits, donde N varía entre 64 y 512 bits, y un número de observaciones L que oscila entre 5 y 10.000.

En este caso X e Y son matrices $L \times N$. Creamos N pares canónicos (U_i, V_i), $1 \leq i \leq N$. Es decir, las matrices U, V son matrices $L \times N$.

En la primera prueba, para un número fijo de observaciones L, variaremos el número de variables N y observaremos cómo se comporta la correlación r_i entre los pares canónicos (U_i, V_i).

Presentamos de nuevo para cada N las correlaciones $r_1, r_N, r_{\frac{N}{2}}$ entre el primer par canónico (U_1, V_1), el último par canónico (U_N, V_N) y el par canónico intermedio $(U_{\frac{N}{2}}, V_{\frac{N}{2}})$. Además, hemos calculado la media y la desviación típica de las correlaciones.

Usamos los parámetros siguientes: $L = 5000$, y variamos N entre los valores $N = 32, 64, 128, \cdots, 512$.

Mostramos una serie de gráficas que muestran cómo varian cada uno de los valores calculados, así como su media y su desviación típica.

En primer lugar, una gráfica que muestra el comportamiento de los pares canónicos primero, último e intermedio:

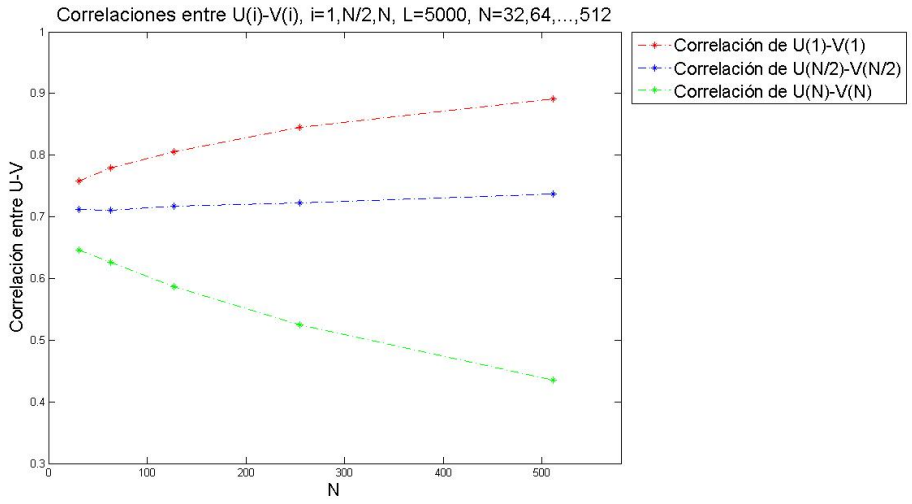

FIGURE 4.9: Correlación de pares canónicos primero, medio y último: $L = 5000$, $N = \{32, 64, 128, \cdots, 512\}$.

Observamos que las correlaciones del primer y último par canónico tienen una diferencia más pronunciada a medida que el número de bits N aumenta. El primer par aumenta su correlación desde 0.75 hasta 0.9, y el último par canónico disminuye desde 0.65 hasta 0.45. El hecho de que haya cada vez mayor diferencia entre las correlaciones de estos dos pares canónicos, se debe a que a medida que aumenta N, la correlación del primer par canónico aumenta debido a que se realiza una combinación lineal de un mayor número de variables, para el mismo número de observaciones. Sin embargo, el último par canónico disminuye, ya que ésta tiene muchas más condiciones de perpendicularidad.

Por otra parte, la correlación del par canónico intermedio se mantiene estable en el intervalo $[0.7 - 0.75]$.

A continuación presentamos una gráfica que representa la variación de la media y la desviación típica a medida que aumenta el número de bits en el mensaje sin aleatorizar:

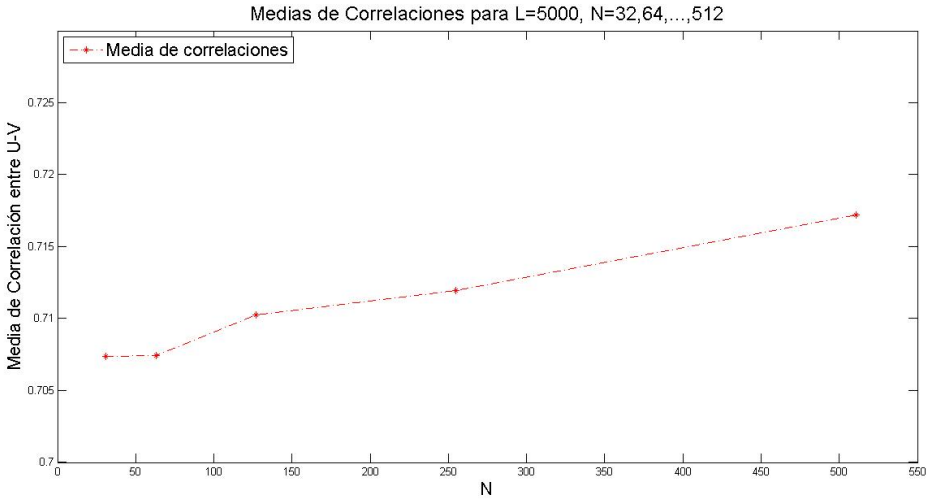

FIGURE 4.10: Media de correlaciones en los pares canónicos: $L = 5000$, $N = \{32, 64, 128, \cdots, 512\}$.

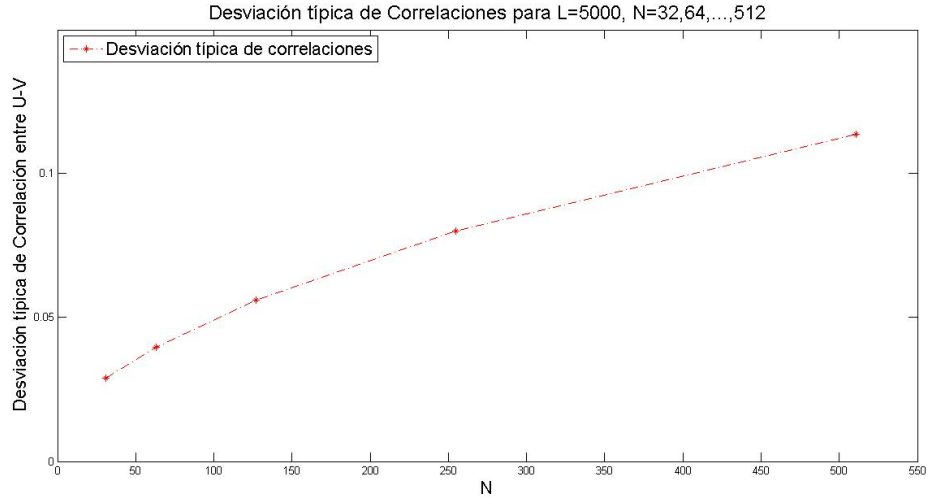

FIGURE 4.11: Desviación típica de correlaciones en los pares canónicos: $L = 5000$, $N = \{32, 64, 128, \cdots, 512\}$.

Observamos que la media se mantiene en general estable aunque crece ligeramente con una estructura exponencial en el rango $[0.705 - 0.72]$.

La desviación típica también crece, aunque de forma logarítmica desde 0.025 hasta 0.125.

En la segunda prueba, observamos qué sucede cuando variamos el número de observaciones L, y dejamos fijo el número de bits N.

De nuevo, vamos a representar el comportamiento de las correlaciones $r_1, r_N, r_{\frac{N}{2}}$ que se dan en el primer par canónico (U_1, V_1), el último par canónico (U_N, V_N) y el par canónico intermedio $(U_{\frac{N}{2}}, V_{\frac{N}{2}})$. Además, hemos calculado la media y desviación típica de las respectivas correlaciones.

Para poder representar el comportamiento, hemos elegido como número de bits $N = 512$, y el número de observaciones variando entre $L = 1000 - 10000$.

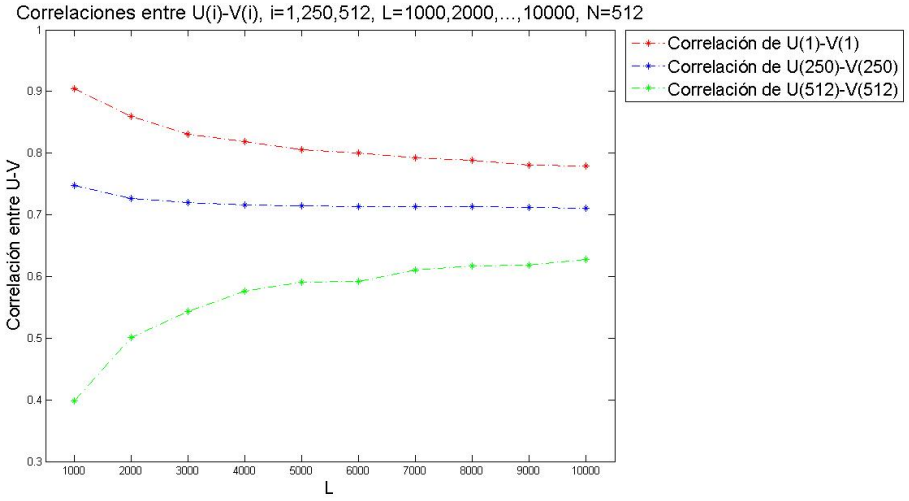

FIGURE 4.12: Correlación de pares canónicos primero, medio y último: $L = \{1000i, i = \{1 \cdots 10\}\}$, $N = 512$.

Observamos que las correlaciones de los pares canónicos primero e intermedio decrecen ligeramente, aunque mantienen rangos superiores a 0.7. El primer par canónico decrece desde 0.9, mientras que el par canónico intermedio lo hace desde 0.75.

La correlación del último par crece de manera logarítimica en el intervalo $[0.4 - 0.65]$.

A continuación vamos a analizar el comportamiento tanto de la media y la desviación típica de la correlación entre los mensajes en claro y los mensajes aleatorizados, aumentando el número de observaciones L y manteniendo el número de bits N tanto de los mensajes en claro como de los mensajes tras la función de aleatorización.

En las gráficas que mostramos a continuación observamos que tanto la media como la desviación típica bajan de forma exponencial, aunque la media se mantiene en el rango $[0.7 - 0.722]$, y la desviación típica en el rango $[0 - 0.14]$:

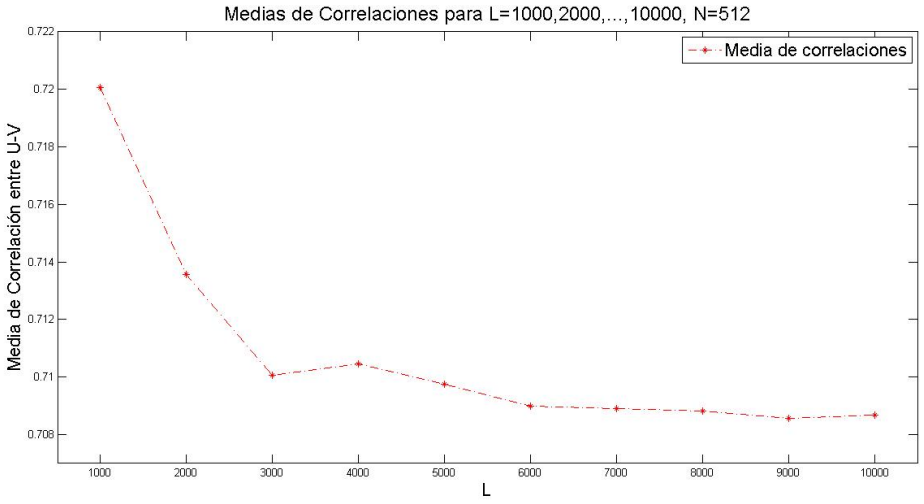

FIGURE 4.13: Media de correlaciones en los pares canónicos: $L = \{1000i, i = \{1 \cdots 10\}\}$, $N = 512$.

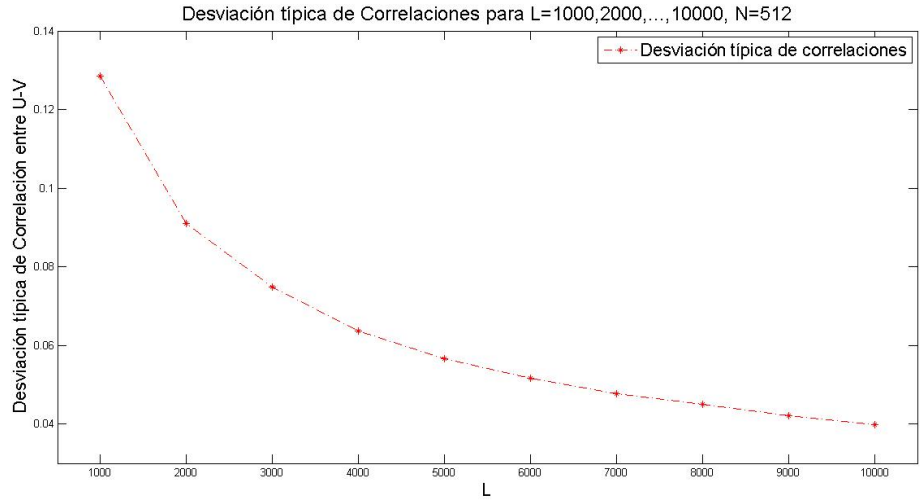

FIGURE 4.14: Desviación típica de correlaciones en los pares canónicos: $L = \{1000i, i = \{1 \cdots 10\}\}$, $N = 512$.

En la última prueba, nos interesamos por saber lo restrictivo que es el Análisis de Correlación Canónica al establecer las condiciones de que perpendicularidad en cada paso.

Para ello, representamos cómo se comporta la correlación a lo largo de los pares canónicos, para $N = 128$, $L = 1000$:

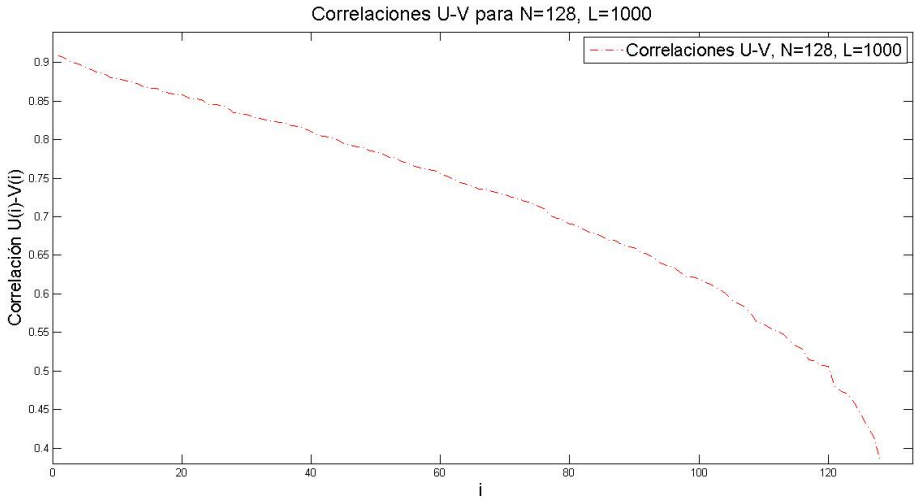

FIGURE 4.15: Correlaciones entre $U(i) - V(i)$: $i = \{1 \cdots 128\}$, $N = 128$, $L = 1000$.

4.5 Conclusiones

El primer algoritmo tenía el objetivo de representar la naturaleza y la potencia del Análisis de Correlación Canónica. Al crear muchas observaciones que consisten en añadir bits aleatorios en determinadas posiciones fijadas al comienzo, dado que el mensaje siempre reside en el mismo lugar, al proyectar el texto aleatorizado sobre las coordenadas del mensaje se maximiza la correlación (ya que se obtiene el texto original). Al aplicar el CCA, los pesos de los bits que corresponden a bits del texto original aumentan, y los pesos de los bits aleatorios disminuyen.

Si comparamos los resultados obtenidos entre la Función Cuadrática y el One Time Pad, podemos observar que el One Time Pad logra disminuir la correlación moderadamente. Sin embargo, no logra una disminución similar al de la Función Cuadrática. Esto se debe por una parte a que el One Time Pad solo dispone de los N bits para ocultar la correlación, y realiza operaciones a nivel de bit. Sin embargo, la Función Cuadrática al disponer de $2N$ bits, tiene espacio suficiente para ocultar la correlación de manera efectiva presentando una estructura no lineal, que opera sobre los valores de los números naturales.

Capítulo 5

Cifrado: Protocolo de distribución de claves

5.1 Introducción

De acuerdo al esquema propuesto en la figura 1.1, tras realizar el proceso de aleatorización, para que la publicación de los pares $(foto, etiqueta)$ sea segura, usamos un esquema de cifrado simétrico. Obsérvese que no nos interesa que todo el mundo pueda cifrar, es por ello que no usamos un criptosistema asimétrico como puede ser RSA o ElGamal. Sin embargo, nos interesamos por que determinados grupos de amigos, familiares, etc. tengan acceso a las etiquetas. Es decir, queremos que estas personas dispongan de la clave para descifrar las etiquetas y ver su contenido.

Por tanto, debemos resolver uno de los problemas principales de los esquemas de cifrado simétrico: el intercambio seguro de claves. Para ello, generalizaremos el conocido protocolo de distribución de claves propuesto por Diffie-Hellman. Además, nos preocuparemos por definir operaciones de adición y eliminación de miembros en el grupo establecido.

El intercambio de claves de forma segura ha sido un problema estudiado desde el inicio de la Criptografía. A lo largo de los años, se han empleado este tipo de protocolos generalmente como medio para acordar claves simétricas.

En nuestro caso vamos a usar una generalización de la distribución de claves Diffie-Hellman. Éste es un protocolo que sirve para establecer claves entre miembros que no han estado en contacto previo, y que se comunican utilizando un canal inseguro.

El esquema fue publicado originalmente por Whitfield Diffie y Martin Hellman en 1976, a pesar de que fue inventado de forma independiente unos años antes en el GCHQ, la

agencia de inteligencia de señales británica, por Malcolm J. Williamson, pero se mantuvo en secreto.

En su versión más simple se tiene que dos personas, Alice y Bob, quieren compartir una clave secreta por el canal inseguro.

Para ello siguen los siguientes pasos:

- Alice y Bob acuerdan operar sobre un grupo finito cíclico G de orden n y un generador $g \in G$.
- Alice elige un número privado aleatorio $1 \leq a \leq n-1$. Ahora calcula $g^a \in G$ y transmite este último valor a Bob.
- De forma simétrica, Bob elige un número privado aleatorio $1 \leq b \leq n-1$, y calcula y envía $g^b \in G$ a Alice.
- Alice recibe g^b y calcula la clave $K = (g^b)^a$.
- Bob recibe g^a y calcula la misma clave $K = (g^a)^b$.

Observamos que el protocolo en sí no es autenticado, por lo que es vulnerable a ataques del tipo Man-in-the-middle [A]. Sin embargo, es fácilmente generalizable a protocolos autenticados.

En nuestro caso, hemos optado por realizar una generalización del protocolo Diffie-Hellman para un grupo de n miembros.

Este grupo se podrá organizar de manera tradicional, es decir, en el que cada uno de los miembros se puede comunicar sin problemas con cualquier otro miembro:

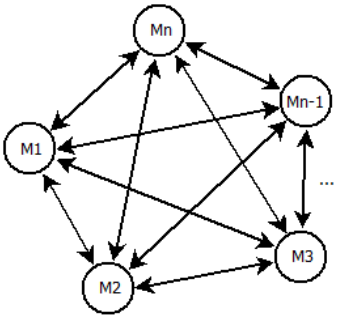

FIGURE 5.1: Topología normal

O bien pueden seguir una topología centralizada. Esto es, todos los miembros del grupo se comunican únicamente con un miembro en particular, que juega el papel de distribuidor:

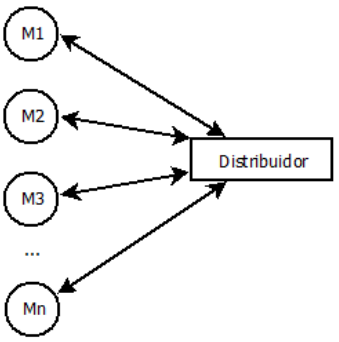

FIGURE 5.2: Topología centralizada

5.2 Protocolo de distribución de claves centralizado

A continuación presentamos un protocolo para realizar el intercambio de claves de forma segura usando la misma filosofía que utilizó Diffie-Hellman pero para el caso en que tenemos n participantes, y una persona que distribuye la información de forma centralizada.

Realizaremos un estudio detallado de la complejidad y la seguridad del protocolo, y estudiaremos también la complejidad de añadir y eliminar miembros al grupo en cuestión.

Finalmente generalizaremos el protocolo para definir subgrupos de miembros.

5.2.1 Notación

Definimos la siguiente notación:

- G es un grupo algebraico. En nuestro protocolo supondremos que G es un cuerpo.
- n es el número de participantes que van a disponer de la clave común.
- A es el distribuidor de la información centralizada.
- M_i es el i-ésimo miembro del grupo, $i \in [1, n]$.
- q es el orden del grupo G.
- α es un generador de G.
- N_i, clave privada de M_i, $i \in [1, n]$.
- N_a, clave privada de A
- $\overline{N_i}$: N_i no incluido
- K clave generada.

5.2.2 Definición del protocolo

Definimos cinco fases en el protocolo.

Paso1

En la primera fase, A colecciona todas las contribuciones de todos los miembros del grupo. Cada miembro M_i computa $\alpha^{N_1 \cdots N_i}$ y envía dicho valor a A.

Entonces, A recibe el valor anterior y lo reenvía a M_{i+1}.

Al final del proceso, M_n calcula el valor $\alpha^{N_1 \cdots N_n}$ y se lo envía a A.

A almacenará este último valor para poder realizar operaciones de adición de miembros, como se explicará en la siguiente subsección.

Paso2

En la segunda fase, A envía el valor recibido por el último miembro $\alpha^{N_1 \cdots N_n}$ por broadcast. Es decir, a todos los miembros del conjunto. Realmente solo sería necesario enviarlo a los miembros $M_1, \cdots, M_{n-1}$, ya que el último de los miembros ya dispone del valor especificado.

Paso3

Durante la tercera fase cada miembro M_i calcula la inversa de su clave privada: N_i^{-1}, y eleva el valor recibido $\alpha^{N_1 \cdots N_n}$ a este valor, obteniendo: $\alpha^{N_1 \cdots \overline{N_i} \cdots N_n}$, y envía este valor a A.

Paso4

En la cuarta fase, A eleva todos los valores recibidos por su clave privada N_a y responde a cada miembro M_i con el valor $\alpha^{N_1 \cdots \overline{N_i} \cdots N_n N_a}$.

A almacenará los valores $\alpha^{N_1 \cdots \overline{N_i} \cdots N_n N_a}$ recibidos de cada miembro $M_i, i \in [1, n]$, con el objetivo de poder eliminar miembros del grupo de forma más eficiente. La razón de esto quedará explícita en la siguiente subsección.

Paso5

Finalmente, en el último paso, cada miembro M_i computa la clave que queremos compartir: $K = \alpha^{N_1 \cdots N_n N_a}$.

A continuación mostramos un esquema general del funcionamiento del protocolo:

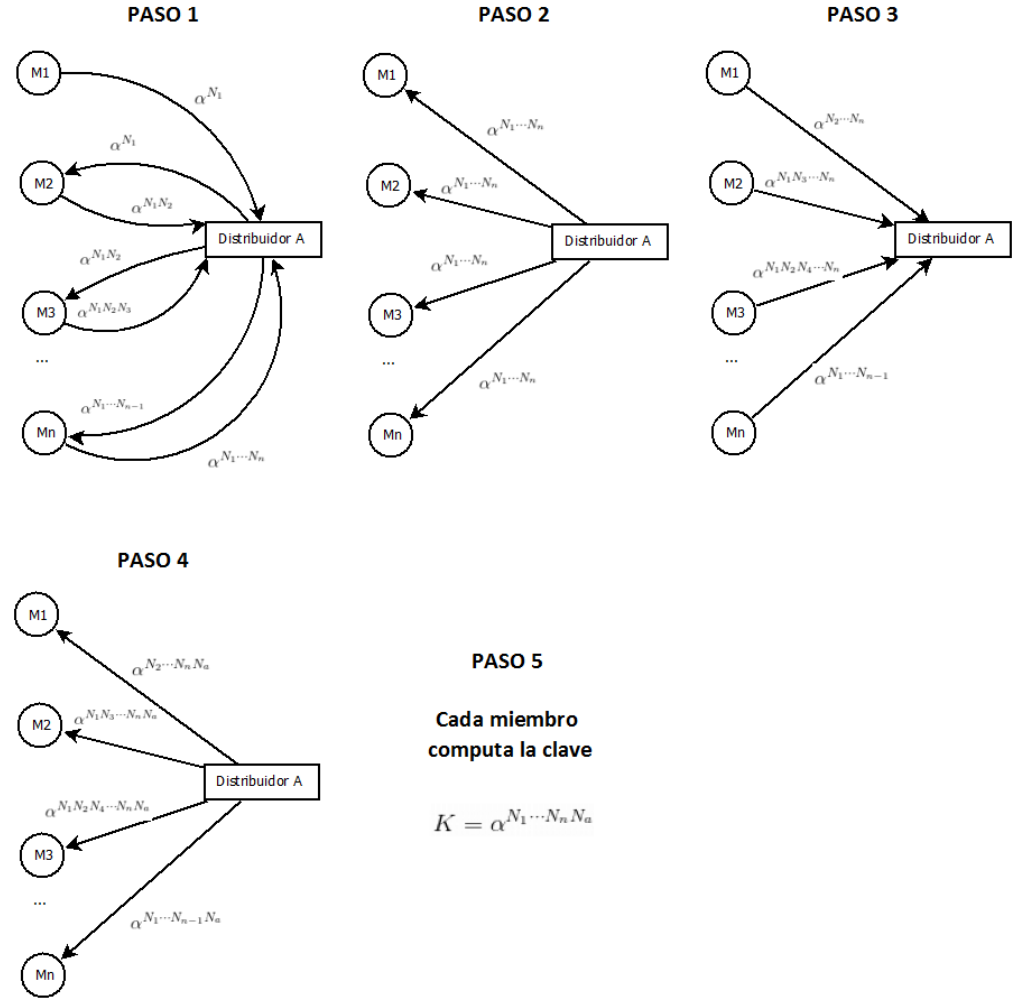

FIGURE 5.3: Protocolo de distribución de claves centralizado

5.2.3 Modificando el grupo

Una vez que hemos definido el grupo de miembros privilegiados que disponen de la clave para poder leer el contenido de los mensajes publicados por el distribuidor, una operación natural es el de querer añadir nuevos miembros al grupo, o querer eliminar miembros del grupo.

5.2.3.1 Añadir un miembro

A continuación vamos a describir cómo añadir un nuevo miembro al grupo previamente definido.

Sea el nuevo miembro M_{n+1}, que posee su clave privada N_{n+1}.

Para realizar el proceso de añadir este nuevo miembro, se realizarán los siguientes pasos:

- A crea una nueva clave privada N_{aa}.
- A envía el valor $\alpha^{N_1 \cdots N_n}$ al nuevo miembro M_{n+1}. Observamos que A había almacenado el valor $\alpha^{N_1 \cdots N_n}$ en el protocolo descrito anteriormente, por lo que puede realizar esta operación.
- M_{n+1} añade su clave privada al valor obtenido y le envía a A $\alpha^{N_1 \cdots N_n N_{n+1}}$.
- A envía un mensaje broadcast a todos los miembros del grupo, con el valor recibido $\alpha^{N_1 \cdots N_n N_{n+1}}$. De nuevo, en este paso destacamos que no es necesario enviar el valor al nuevo miembro M_{n+1}, ya que éste ya tiene dicho valor.
- Cada miembro M_i envía el valor $\alpha^{N_1 \cdots \overline{N_i} \cdots N_n N_{n+1}}$ a A elevando el valor anterior por la inversa de su clave privada, como en el paso 3 del protocolo de distribución de claves centralizado.
- Repetimos el paso 4 del protocolo, respondiendo a cada miembro $M_i, i \in [1, n+1]$ con el valor correspondiente $\alpha^{N_1 \cdots \overline{N_i} \cdots N_n N_{n+1} N_{aa}}$.

  Observamos en este paso que elevamos los valores obtenidos a la nueva clave privada N_{aa}, y no a la clave privada anterior N_a.

  Si no fuera así, el protocolo tendría dos serias vulnerabilidades: el nuevo miembro podría obtener la clave del grupo anterior. Y peor aún, el distribuidor A habría enviado por el canal la clave anterior $\alpha^{N_1 \cdots \overline{N_i} \cdots N_n N_{n+1} N_{aa}}$ al nuevo miembro M_{n+1}, y un usuario desconocido podría haber espiado y obtenido la clave del grupo anterior, pudiendo revelar todos los mensajes compartidos hasta ese momento.
- Cada miembro del grupo computa la nueva clave $K = \alpha^{N_1 \cdots N_n N_{n+1} N_{aa}}$.

5.2.3.2 Eliminar un miembro

En primer lugar observamos que el distribuidor A guardó los valores $\alpha^{N_1 \cdots \overline{N_i} \cdots N_n N_a}$ recibidos de cada miembro $M_i, i \in [1, n]$.

Supongamos que el miembro M_i ha cometido un serio delito y es retirado del grupo, o ha decidido abandonar el grupo por sí mismo.

Para realizar la operación de eliminar un miembro del grupo, se realizarán los siguientes pasos:

- A genera una nueva clave privada N_{aa}, y envía por broadcast (a todos los miembros del grupo) el valor $\alpha^{N_1 \cdots \overline{N_i} \cdots N_n}$, que es uno de los valores que A había guardado durante el proceso de distribución de la primera clave.

- Cada miembro $M_j, j \neq i$, calcula la inversa de su clave privada y envía el valor $\alpha^{N_1 \cdots \overline{N_i} \cdots \overline{N_j} \cdots N_n}$ a A.
- A añade su nueva clave privada y responde a cada M_j con el nuevo valor:

$$\alpha^{N_1 \cdots \overline{N_i} \cdots \overline{N_j} \cdots N_n N_{aa}}$$

- Finalmente, cada miembro computa la nueva clave: $\alpha^{N_1 \cdots \overline{N_i} \cdots N_n N_{aa}}$.

Observamos que el hecho de que A genere una nueva clave privada es primordial. Si no fuera así, el antiguo miembro del grupo M_i podría acceder a la información privilegiada del grupo, a pesar de que ya no es un miembro.

5.2.4 Generalización

El escenario generalizado es el siguiente: supongamos que el distribuidor A quiere transmitir mensajes distintos a distintos subgrupos del grupo de n personas. Queremos saber cómo podemos afrontar el problema de que A quiera enviar un mensaje m_i solamente a un subconjunto de personas disponibles, pero no a las demás, y un mensaje m_j a otro subconjunto de personas, que pueden tener o no miembros en común.

En este caso, el protocolo se modificará de la siguiente manera:

- Los pasos 1, 2 y 3 se reproducen exactamente igual que en el protocolo establecido anteriormente.
- En el paso 4, A genera tantas claves privadas N_{A_i} como subgrupos quiera definir. Supongamos que quiere definir l subgrupos siendo los subgrupos $S = \{s_1, s_2, \cdots, s_l\}$. Entonces enviará cada valor recibido por el miembro $M_j, \alpha^{N_1 \cdots \overline{N_j} \cdots N_n}$, elevado a las claves N_{a_i} correspondientes a los subgrupos a los que pertenezca el miembro M_j. Es decir, si M_j está en los subgrupos $s_a, s_b, s_c, a, b, c \in [1, l]$, entonces A calculará y enviará a M_j los valores $\alpha^{N_1 \cdots \overline{N_j} \cdots N_n N_{A_a}}, \alpha^{N_1 \cdots \overline{N_j} \cdots N_n N_{A_b}}, \alpha^{N_1 \cdots \overline{N_j} \cdots N_n N_{A_c}}$.
- En el último paso, cada miembro M_i computa las claves de los subgrupos a los que pertenece simplemente elevando a su propia clave privada el valor recibido por A en el paso anterior.

5.2.4.1 Ejemplo

Supongamos que además del distribuidor A hay 4 participantes($n = 4$), y que el distribuidor dispone de 3 mensajes ($l = 3$): m_1, m_2, m_3. Si por ejemplo A quiere enviar

el mensaje m_1 a M_1, M_2, el mensaje m_2 a M_1, M_3 y el mensaje m_3 a M_2, M_3, M_4, se procederá de la siguiente forma:

- Paso 1: M_1 manda el valor α^{N_1} a A. A recibe el valor y lo manda a M_2. M_2 manda el valor $\alpha^{N_1 N_2}$ a A. A recibe el valor y lo manda a M_3. M_3 manda el valor $\alpha^{N_1 N_2 N_3}$ a A. A recibe el valor y lo manda a M_4. M_4 manda el valor $\alpha^{N_1 N_2 N_3 N_4}$ a A, y éste lo guarda.
- Paso 2: A envía el valor $\alpha^{N_1 N_2 N_3 N_4}$ a todos los miembros.
- Paso 3: Cada miembro $M_i, i \in [1, 4]$ calcula la inversa de su clave privada: N_i^{-1}, y eleva el valor recibido $\alpha^{N_1 \cdots N_4}$ a este valor, obteniendo: $\alpha^{N_1 \cdots \overline{N_i} \cdots N_4}$, y envía este valor a A. Es decir, M_1, M_2, M_3, M_4 envían los valores $\alpha^{N_2 N_3 N_4}$,$\alpha^{N_1 N_3 N_4}$,$\alpha^{N_1 N_2 N_4}$, $\alpha^{N_1 N_2 N_3}$ a A respectivamente, y éste almacena todos los valores recibidos.
- Paso 4: A define los tres subgrupos: $S_1 = \{M_1, M_2\}, S_2 = \{M_1, M_3\}, S_3 = \{M_2, M_3, M_4\}$, y crea tres claves privadas $N_{A_1}, N_{A_2}, N_{A_3}$, una para cada subgrupo. Entonces, para el primer subgrupo calcula el valor $\alpha^{N_2 N_3 N_4 N_{A_1}}$ y lo manda a M_1, y calcula $\alpha^{N_1 N_3 N_4 N_{A_1}}$ y lo manda a M_2. Para el segundo subgrupo, mandará los valores $\alpha^{N_2 N_3 N_4 N_{A_2}}$, $\alpha^{N_1 N_2 N_4 N_{A_2}}$ a M_1, M_3 respectivamente. Asimismo, para el tercer y último subgrupo, mandará los valores $\alpha^{N_1 N_3 N_4 N_{A_3}}$, $\alpha^{N_1 N_2 N_4 N_{A_3}}$, $\alpha^{N_1 N_2 N_3 N_{A_3}}$ a los miembros M_2, M_3, M_4 respectivamente.
- Paso 5: Cada miembro del grupo es capaz de computar la clave que le permitirá descifrar el contenido del mensaje de cada subgrupo. En concreto: M_1 recibe los valores $\alpha^{N_2 N_3 N_4 N_{A_1}}$, $\alpha^{N_2 N_3 N_4 N_{A_2}}$, por lo que es capaz de calcular las claves $K_1 = \alpha^{N_1 N_2 N_3 N_4 N_{A_1}}, K_2 = \alpha^{N_1 N_2 N_3 N_4 N_{A_2}}$ que le permitirán ver los mensajes m_1, m_2, ya que forma parte de los subgrupos S_1 y S_2. De la misma forma, M_2 es capaz de calcular las claves K_1 y K_3, M_3 las claves K_2 y K_3, y finalmente M_4 solamente la clave K_3, ya que el único valor que recibe por parte de A es $\alpha^{N_1 N_2 N_3 N_{A_3}}$, por lo que solamente puede calcular $K_3 = \alpha^{N_1 N_2 N_3 N_4 N_{A_3}}$.

Para añadir y eliminar miembros de cada subgrupo, simplemente se procede como en el protocolo no generalizado, tratando al subgrupo en cuestión como si fuera un grupo entero e independiente.

5.2.5 Complejidad

A continuación se expone la complejidad computacional tanto para el protocolo, como para las operaciones de adición y borrado de miembros, y también para el caso del protocolo generalizado.

Protocolo centralizado no generalizado En el paso 1, A recolecta las contribuciones de cada miembro, y se transmiten un total de $2n$ claves. En el paso 2 A realiza un broadcast, por lo que se transmiten n claves. En el paso 3, cada miembro responde a A quitando su clave privada, por lo que se transmiten n claves de nuevo. En el paso 4, A eleva los valores recibidos a su clave privada N_a, y responde a cada miembro del grupo, por lo que se transmiten n claves de nuevo. En el paso 5 cada miembro computa la clave compartida, por lo que no se transmiten claves.

En conclusión, este proceso requiere un total de $5n$ transmisiones de claves. Es un coste lineal sobre el número de miembros del grupo.

Adición de un miembro En el paso 1, A se comunica con el nuevo miembro del grupo, y se realizan 2 transmisiones. Los siguientes pasos son similares a los pasos 2,3,4 del protocolo, por lo que el coste es $3n$. Esto lleva a realizar un total de $3n + 2$ transmisiones.

Eliminar un miembro En el primer paso se manda un valor a todos los miembros menos al eliminado, por lo que se realizan $(n - 1)$ transmisiones. En el segundo paso cada miembro responde a A, realizando un total de $(n - 1)$ transmisiones de nuevo. Cuando A responde, realiza de nuevo $(n - 1)$ transmisiones.

En total, se realizan $3(n - 1)$ transmisiones para el proceso de borrado.

Protocolo centralizado generalizado Los primeros 3 pasos se consumen $4n$ transmisiones, al igual que el protocolo. En el paso 4, se envía a cada miembro de cada subgrupo una subclave. Si un miembro no aparece en ningún subgrupo no se le manda ninguna clave, al igual que si un miembro aparece en repetidos subgrupos, se enviarán diferentes subclaves a un mismo miembro. Por tanto, si los subgrupos son $S = \{s_1, s_2, \cdots, s_l\}$, se realizan $\sum_i |s_i|$, donde $|s_i|$ es el cardinal del subgrupo i.

A modo de resumen, presentamos la siguiente tabla:

Operación	Transmisión de claves
Básico	$5n$
Añadir	$3n + 2$
Eliminar	$3n - 3$
Generalizado	$4n + \sum_i \|s_i\|$

TABLE 5.1: Complejidad Protocolo Centralizado y Operaciones

5.3 Protocolo de distribución de claves no centralizado

Supongamos que un grupo de amigos quiere distribuirse mensajes entre ellos, sin que haya un distribuidor de mensajes central. En este caso deciden que todos pueden transmitir mensajes a todos. Además, disponen de medios para poder comunicarse entre todos.

Queremos saber cómo podemos realizar la distribución de claves de forma segura bajo esta topología.

5.3.1 Definición del protocolo

De nuevo, definimos cinco fases en el protocolo.

Paso1

La primera fase es similar a la del protocolo centralizado. Sin embargo, ahora no necesitamos que las claves pasen por el distribuidor A, ya que ahora todos son distribuidores de mensajes, por lo que el coste será menor.

Entonces, cada miembro M_i computa $\alpha^{N_1 \cdots N_i}$ y envía dicho valor directamente a M_{i+1}.

Este proceso se realizará hasta M_{n-1}, no hasta el miembro M_n.

Paso2

M_{n-1} realiza un broadcast mandando el valor $\alpha^{N_1 \cdots N_{n-1}}$ a todos los miembros del grupo.

Podemos observar que en este momento M_n recibe el valor de M_{n-1}: $\alpha^{N_1 \cdots N_{n-1}}$, y puede computar la clave $K = \alpha^{N_1 \cdots N_{n-1} N_n}$.

Paso3

Durante la tercera fase cada miembro $M_i, i \in [1, n-1]$ calcula la inversa de su clave privada: N_i^{-1}, y eleva el valor recibido por M_{n-1} a la inversa de su clave, obteniendo: $\alpha^{N_1 \cdots \overline{N_i} \cdots N_{n-1}}$, y envía este valor a M_n.

Paso4

En la cuarta fase, M_n eleva todos los valores recibidos por su clave privada N_n y responde a cada miembro $M_i, i \in [1, n-1]$ con el valor $\alpha^{N_1 \cdots \overline{N_i} \cdots N_n}$.

Paso5

Finalmente, en el último paso, cada miembro $M_i, i \in [1, n-1]$ computa la clave que queremos compartir: $K = \alpha^{N_1 \cdots N_n}$.

A continuación mostramos un esquema general del funcionamiento del protocolo:

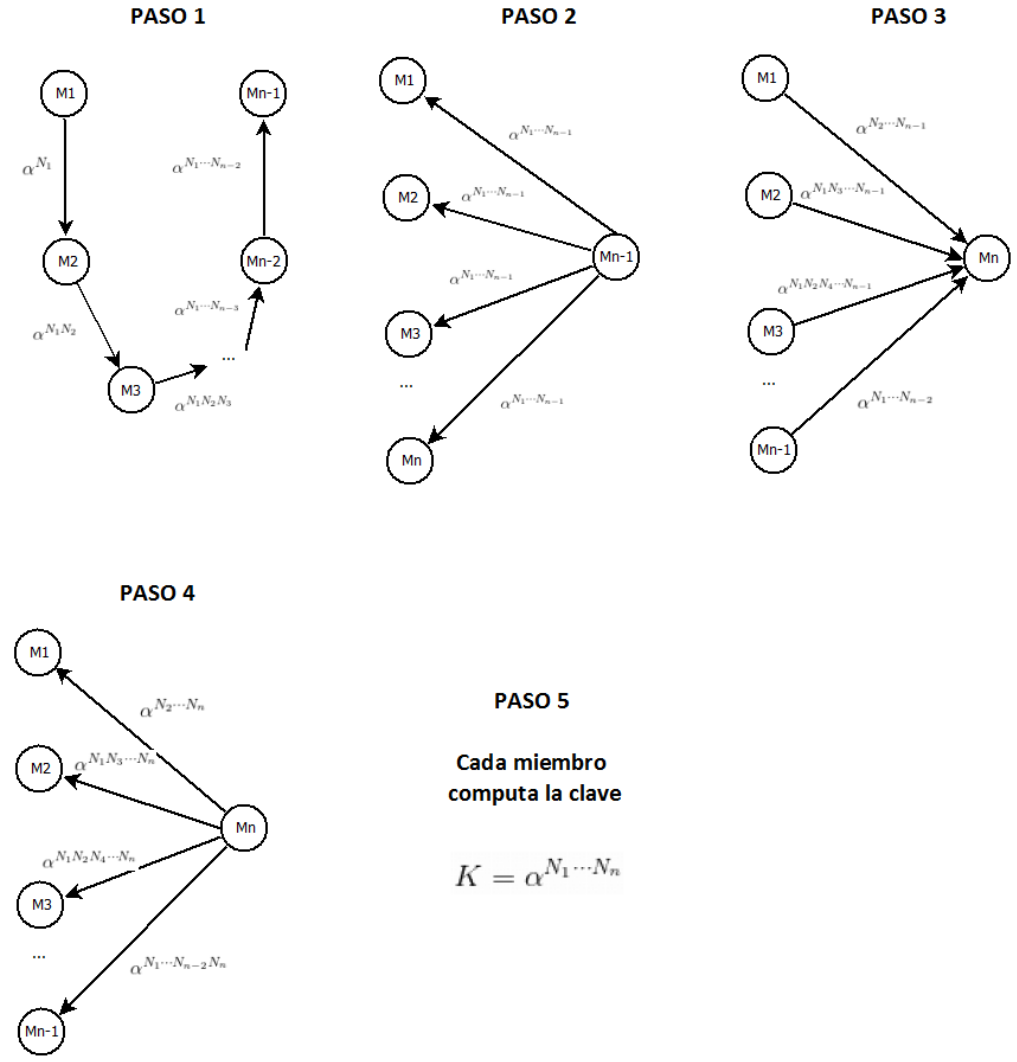

FIGURE 5.4: Protocolo de distribución de claves no centralizado

5.3.2 Modificación del grupo

De nuevo, habiendo definido el grupo de miembros privilegiados que disponen de la clave para poder leer el contenido de los mensajes publicados por el distribuidor, queremos de nuevo añadir o eliminar miembros al grupo.

5.3.2.1 Añadir un miembro

Designamos al nuevo miembro como M_{n+1} y a su clave privada como N_{n+1}.

Se realizarán pasos similares a los del protocolo centralizado:

- M_n crea una nueva clave privada N_{nn}.
- M_n envía el valor $\alpha^{N_1 \cdots N_{n-1}}$ al nuevo miembro M_{n+1}.
- M_{n+1} añade su clave privada al valor obtenido y le envía a M_n $\alpha^{N_1 \cdots N_{n-1} N_{n+1}}$.
- M_n envía un mensaje broadcast a todos los miembros del grupo, con el valor recibido $\alpha^{N_1 \cdots N_{n-1} N_{n+1}}$.
- Cada miembro $M_i, i \neq n$ envía el valor $\alpha^{N_1 \cdots \overline{N_i} \cdots N_n N_{n+1}}$ a M_n elevando el valor anterior por la inversa de su clave privada.
- Respondemos a cada miembro $M_i, i \in [1, n+1]$ con el valor correspondiente $\alpha^{N_1 \cdots \overline{N_i} \cdots N_{nn} N_{n+1}}$.
- Cada miembro del grupo computa la nueva clave $K = \alpha^{N_1 \cdots N_{nn} N_{n+1}}$.

5.3.2.2 Eliminar un miembro

Los pasos a realizar son similares a los del protocolo centralizado:

- M_n genera una nueva clave privada N_{nn}, y envía por broadcast (a todos los miembros del grupo) el valor $\alpha^{N_1 \cdots \overline{N_i} \cdots N_{n-1}}$.
- Cada miembro $M_j, j \neq i$, calcula la inversa de su clave privada y envía el valor $\alpha^{N_1 \cdots \overline{N_i} \cdots \overline{N_j} \cdots N_{n-1}}$ a M_n.
- M_n añade su nueva clave privada y responde a cada M_j con el nuevo valor: $\alpha^{N_1 \cdots \overline{N_i} \cdots \overline{N_j} \cdots N_{n-1} N_{nn}}$.
- Finalmente, cada miembro computa la nueva clave: $\alpha^{N_1 \cdots \overline{N_i} \cdots N_{n-1} N_{nn}}$.

5.3.3 Complejidad

A continuación se estudia la complejidad computacional para el protocolo no centralizado y las operaciones de modificación.

Protocolo no centralizado M_{n-1} recolecta las contribuciones de cada miembro, transmitiéndose $n-2$ claves. En el paso 2 M_{n-1} realiza un broadcast, por lo que se transmiten $n-1$ claves. En el paso 3, cada miembro responde a M_n quitando su clave privada, por lo que se transmiten $n-1$ claves de nuevo. En el paso 4, M_n eleva los valores recibidos

a su clave privada N_n, y responde a cada miembro del grupo, por lo que se transmiten $n-1$ claves de nuevo. En el paso 5 cada miembro computa la clave compartida, por lo que no se transmiten claves.

El coste total es de $4n-5$ claves.

Adición y Eliminación de un miembro El coste de añadir un miembro es de $3n+2$ transmisiones de clave, como en el protocolo centralizado. Y el coste de eliminar un miembro es de $3(n-1)$ transmisiones para el proceso de borrado.

A modo de resumen, presentamos la siguiente tabla:

Operación	Transmisión de claves
No centralizado	$4n-5$
Añadir	$3n+2$
Eliminar	$3n-3$

TABLE 5.2: Complejidad del protocolo no centralizado y operaciones

5.4 Justificación de la seguridad de los protocolos centralizado y no centralizado

A continuación justificamos la seguridad de los protocolos y las operaciones de modificación del grupo que hemos descrito anteriormente.

Para ello, en primer lugar tenemos que definir en qué se basa la seguridad de los protocolos, y es en el problema del Logaritmo Discreto y el problema de Diffie-Hellman.

5.4.1 El Problema del Logaritmo Discreto

Dado un grupo G que tiene orden q, en nuestro caso primo, un elemento $h \in G$ y un generador g, el Problema del Logaritmo Discreto consiste en encontrar el exponente $\alpha = log_g h$ tal que $g^\alpha = h$.

Se dice que un algoritmo resuelve el Problema del Logaritmo Discreto si, para las entradas $g \neq 1$, y un elemento $h \in G$ generado de manera aleatoria, devuelve el $log_g h$ con por lo menos una probabilidad de éxito mayor a una constante positiva.

La suposición del Logaritmo Discreto establece que no existe ningún algoritmo de tiempo polinomial sobre el número de bits que resuelva el Problema del Logaritmo Discreto con una probabilidad de éxito grande.

Esta suposición es equivalente a asumir que no existen un generador $g \in G$ y un algoritmo polinomial tal que dada una entrada aleatoria $h \in G$, el algoritmo devuelva $log_g h$ con una probabilidad de éxito grande.

Proposición: Las dos siguientes afirmaciones son equivalentes:

1. Existe un algoritmo de tiempo polinomial $A_{(a)}$ que, dadas las entradas aleatorias $g \neq 1$ y h, devuelve $log_g h$.
2. Existen un elemento $g \neq 1$ y un algoritmo polinomial $A_{(b)}$ que, recibiendo una entrada aleatoria h, devuelve $log_g h$.

Demostración – . La implicación $1 \rightarrow 2$ es obvia. Para probar la otra implicación supongamos que tenemos las entradas g_1, h, y que queremos calcular $log_{g_1} h$. El algoritmo $A_{(a)}$, en primer lugar introduce g_1 en $A_{(b)}$ para calcular para calcular $a = log_g g_1$. Entonces, introduce h en $A_{(b)}$ y calcula $b = log_g h$, devolviendo como solución final $\frac{b}{a} \, mod(q)$.

A día de hoy, aún no se sabe con certeza que la suposición del Logaritmo Discreto (DLA) sea cierta, pero en general se considera una suposición válida.

5.4.2 El Problema de Diffie-Hellman

El problema en su forma más básica consiste en encontrar una clave única Diffie-Hellman de la forma g^{ab} a partir de $g_1 = g^a$ y $g_2 = g^b$.

Un algoritmo se dice que resuelve el Problema Diffie-Hellman sobre la base g si, para las entradas g_1, g_2 generadas de forma aleatoria, devuelve la clave g^{ab} con una probabilidad de éxito no despreciable.

La suposición de Diffie-Hellman Computacional (CDH) consiste en asumir que para $g \neq 1$ no existe ningún algoritmo que resuelva el problema de Diffie-Hellman en tiempo polinomial en el número de bits en la entrada.

La suposición de Diffie-Hellman en su versión Decisiva (DDH), establece que si trabajamos en un grupo cíclico de orden q, con g como generador, las tuplas (g^a, g^b, g^{ab}), donde a y b son elegidos aleatoriamente y de forma independiente de Z_q, y (g^a, g^b, g^c), donde a, b, c son elegidos de forma independiente de Z_q, son computacionalmente indistinguibles.

En términos de la Complejidad Computacional, dos variables aleatorias X, Y son computacionalmente indistinguibles si para cualquier algoritmo de tiempo polinomial A, y cota polinómica ϵ, se tiene:

$$|Pr[A(X) = 1] - Pr[A(Y) = 1]| \leq \epsilon$$

Intuitivamente, quiere decir que las dos secuencias de variables aleatorias se parecen mucho. En nuestro caso, querría decir que g^{ab} se parece mucho a un valor aleatorio del grupo.

Cabe destacar que se considera que esta suposición es más fuerte que la suposición del Logaritmo Discreto y la suposición Diffie-Hellman Computacional.

Por otra parte, si no se cumple DLA, tampoco se cumple DDH porque se podría calcular $log_g g^a$, y comprobar si nuestro g^c es $(g^b)^a$. Lo mismo pasa con CDH. Si fuera posible calcular de forma eficiente g^{ab} a partir de g^a y g^b, podríamos realizar un procedimiento similar.

5.4.3 Prueba de Seguridad

En el contexto de la indistinguibilidad computacional decimos que el protocolo de Diffie-Hellman es seguro si cumple que es computacionalmente indistinguible. En su versión más simple (distribución de claves entre dos miembros), es claro que si suponemos que se cumple DDH, el protocolo es seguro.

Para probar la seguridad de nuestros protocolos, queremos saber si se mantiene la propiedad de ser computacionalmente indistinguibles cuando extendemos la distribución de claves para un grupo de n personas.

En nuestros protocolos consideramos cuerpos en donde se asume que se cumple la hipótesis DDH. En concreto, podemos suponer que disponemos de un algoritmo que genera de manera aleatoria el par (q, α) donde q tiene k bits, y donde q y $q' = 2q + 1$ son primos. Además, α genera un subgrupo único G de Z_g^* de orden q.

Tras generar el grupo G con su generador α, generamos las claves privadas $X = (N_1, \cdots N_n)$, donde $N_i \in Z_q$.

Definimos:

- $view(q, \alpha, n, X)$:= El conjunto de claves públicas de la forma $\alpha^{N_{i_1} \cdots N_{i_m}}$, donde $i_1 \cdots i_m$ es un subconjunto propio de $1, \cdots n$.

- $K(q, \alpha, n, X) = \alpha^{N_1 \cdots N_n}$, es la clave acordada en el protocolo.

Como el grupo tiene fijado los parámetros q, α, denotamos $view(n, X) = view(q, \alpha, n, X)$ y $K(n, X) = K(q, \alpha, n, X)$.

Además, denotamos:

1. $\approx_{poly}$: indistinguibilidad computacional.
2. $A_n = (view(n, X), y)$, $y \in G$ aleatorio.
3. $D_n = (view(n, X), K(n, X))$

Teorema: Si se cumple DDH (es decir, $A_2 \approx_{poly} D_2$), entonces $A_n \approx_{poly} D_n$ para todo $n \geq 2$.

Demostración − . Procedemos a realizar una demostración por inducción:

Tenemos el caso base $A_2 \approx_{poly} D_2$, y supongamos que se cumple que $A_{n-1} \approx_{poly} D_{n-1}$. Veamos que se cumple también que $A_n \approx_{poly} D_n$.

En primer lugar, observamos que si definimos $X = (N_3, \cdots, N_n)$, podemos definir de manera recursiva $view(n, (N_1, N_2, X))$ como: $view(n, (N_1, N_2, X)) = (view(n - 1, (N_1, X)), K(n-1, (N_1, X), view(n-1, (N_2, X)), K(n-1, (N_2, X), view(n-1, (N_1N_2, X))$.

Por otra parte, $K(n, (N_1, N_2, X)) = K(n - 1, (N_1N_2, X))$.

Entonces, redefiniendo A_n y D_n tenemos:

$A_n = (view(n-1, (N_1, X)), K(n-1, (N_1, X), view(n-1, (N_2, X)), K(n-1, (N_2, X), view(n-1, (N_1N_2, X), y)$.

$D_n = (view(n-1, (N_1, X)), K(n-1, (N_1, X), view(n-1, (N_2, X)), K(n-1, (N_2, X), view(n-1, (N_1N_2, X), K(n-1, (N_1N_2, X)))$.

Ahora definimos B_n, C_n como:

$B_n = (view(n-1, (N_1, X)), K(n-1, (N_1, X), view(n-1, (N_2, X)), K(n-1, (N_2, X), view(n-1, (c, X), y)$.

$C_n = (view(n-1, (N_1, X)), K(n-1, (N_1, X), view(n-1, (N_2, X)), K(n-1, (N_2, X), view(n-1, (c, X), K(n-1, (c, X)))$.

donde c es un elemento aleatorio de Z_q.

Demostremos primero que $A_n \approx_{poly} B_n$:

Supongamos que un adversario distingue entre A_n y B_n. Como $\alpha^{N_1N_2} \in view(n-1,(N_1N_2,X))$, y $\alpha^c \in view(n-1,(c,X))$, entonces seríamos capaces de distinguir entre A_2 y B_2, contradiciendo la hipótesis DDH.

Ahora veamos que $B_n \approx_{poly} C_n$:

De nuevo, supongamos que un adversario distingue entre B_n y C_n. Consideramos el problema de saber si $y = K(n-1,(c,X))$, es decir, una instancia de $A_{n-1} \approx_{poly} D_{n-1}$.

Como $y \in B_n$ y $K(n-1,(c,X)) \in C_n$, entonces podemos distinguir si $y = K(n-1,(c,X))$, y, por tanto, no tendríamos que $A_{n-1} \approx_{poly} D_{n-1}$ que es la hipótesis de inducción.

Finalmente, demostramos que $C_n \approx_{poly} D_n$, de donde se deducirá por transitividad que $A_n \approx_{poly} D_n$:

Si un adversario distingue entre C_n y D_n, puede distinguir entre $\alpha^{N_1N_2} \in view(n-1,(N_1N_2,X)) \subset D_n$, y $\alpha^c \in view(n-1,(c,X)) \subset C_n$, entonces seríamos capaces de distinguir entre A_2 y B_2, contradiciendo la hipótesis DDH como en el primer ejemplo.

Por tanto, $A_n \approx_{poly} D_n$ como queríamos demostrar, por transitividad.

Capítulo 6

Conclusiones

6.1 Resultados obtenidos

El presente trabajo tuvo como objetivo encontrar un esquema efectivo y seguro para prevenir ataques basados en correlación de posibles intrusos en contextos en los que se publiquen un gran conjunto de pares $(dato, metadato)$, como lo son las redes sociales a día de hoy. Por ejemplo, es común encontrar gran cantidad de pares $(foto, etiqueta)$ en la red, debido a la gran cantidad de fotos etiquetadas existentes.

Para ello, en primer lugar nos hemos centrado en diseñar distintos esquemas para romper la posible correlación existente entre los datos y metadatos. La herramienta que hemos usado para medir las correlaciones, dada su potencia y generalidad ha sido el Análisis de Correlación Canónica.

Obsérvese que dado que a priori presuponemos que los datos y metadatos contienen una correlación muy alta, nos basta con aleatorizar los metadatos, que son cadenas de bits pequeñas, para que no conserven correlación.

Hemos observado que esquemas sencillos como la adición de bits en posiciones fijas no disminuyen en absoluto la correlación, debido a la naturaleza del Análisis de Correlación Canónica. Sin embargo, otros esquemas como el One Time Pad o el esquema que hemos diseñado y llamado Función Cuadrática, tienen un efecto positivo a la hora de disminuir la correlación en estos pares.

En concreto, observamos que el One Time Pad disminuye la correlación, pero no obtiene resultados completamente satisfactorios ya que para 10.000 observaciones, la media de las correlaciones entre los pares canónicos se mantiene por encima de 0.7, para cadenas de 512 bits. Sin embargo, hemos comprobado que la función que hemos diseñado, la Función

Cuadrática, baja la correlación hasta casi 0.1. En los dos algoritmos la desviación típica es prácticamente nula usando 10.000 pares de observaciones y suponiendo metadatos de 128 bits.

Por otra parte, para que el esquema sea seguro, usamos la salida obtenida tras aleatorizar como entrada de un esquema de cifrado simétrico seguro como el AES. La persona que quería publicar los pares $(dato, metadato)$, publicaría pares $(dato, c(f(metadato)))$, donde la función f es de aleatorización, y c es una función de cifrado.

En este paso en concreto, teníamos que tener en cuenta que determinadas personas querrían descifrar el contenido de las etiquetas. Para ello, necesitábamos distribuir la clave de manera segura. Ello nos ha llevado a usar el diseño de un protocolo basado en la idea original de Diffie-Hellman, pero extendido a un conjunto arbitrario de miembros. Además, hemos podido definir operaciones de adición y borrado de miembros.

Estos protocolos los hemos diseñado para topologías centralizadas en las que solamente un miembro, el distribuidor, se puede comunicar con las demás personas, y topologías completas, en las que todo par de miembros se pueden comunicar.

Los resultados que hemos obtenido en este protocolo son satisfactorios, ya que a pesar de que la mayoría de generalizaciones del protocolo de Diffie-Hellman requieren un número de intercambios de claves que crece de manera cuadrática, el protocolo propuesto requiere un intercambio de claves de orden lineal. En concreto, $5n$ intercambios de claves para la topología centralizada y $4n$ intercambios para la topología completa o no centralizada.

En conclusión, consideramos que el esquema propuesto resuelve de manera correcta el problema propuesto que queríamos resolver.

6.2 Posibles Trabajos Futuros

Existen diversos puntos en este presente trabajo que se pueden extender como posibles trabajos futuros.

En concreto, al realizar los estudios sobre el Análisis de Correlación Canónica, nos estamos restringiendo a correlaciones lineales entre los datos y metadatos. Cabe destacar que se podrían obtener correlaciones no lineales en dichos pares.

A día de hoy existen otros métodos más avanzados para encontrar otro tipo de correlaciones. Entre ellos el uso de redes neuronales. Se podrían diseñar algoritmos de aleatorización más complejos para evadir correlaciones tanto lineales como no lineales.

Por otra parte, se podrían considerar mejoras sobre la distribución segura de claves. En concreto podríamos considerar otras topologías de miembros, para observar si se puede optimizar más la distribución de claves, así como las operaciones de adición y eliminación de miembros. De la misma forma, se podrían investigar protocolos de distribución de claves alternativos al presentado.

Apéndice A

Man in the Middle

A.1 Man in the Middle

Un ataque Man in the Middle se da en un contexto en el que el intruso malicioso es capaz de leer y modificar los mensajes que se envían dos partes sin que ninguna de ellas sea consciente.

Este ataque es un ejemplo típico que se da en el protocolo de intercambio de claves de Diffie-Hellman, cuando éste no tiene autenticación.

Para ilustrar la vulnerabilidad del protocolo de Diffie-Hellman, vamos a describir los pasos suponiendo que A y B quieren compartir una clave común, y C realiza el ataque:

1. A intenta enviar su clave pública g^a a B.
2. C intercepta la clave de A, y envía en su lugar su propia clave g^c a B.
3. B recibe g^c pensando que viene de A, y contesta con g^b.
4. C intercepta la clave de B y en su lugar envía a A g^c.
5. A computa una clave compartida con C (g^{ac}), y B computa una clave compartida con C (g^{bc}). Mientras, C tiene ambas claves.

De esta manera, cuando por ejemplo A envíe un mensaje m a B, C podrá interceptar el mensaje, descifrarlo con la clave que comparte con A para ver el contenido, y cifrarlo con la clave que comparte con B, para enviárselo a éste sin que levante ningún tipo de sospecha.

A continuación mostramos un esquema que resume los pasos:

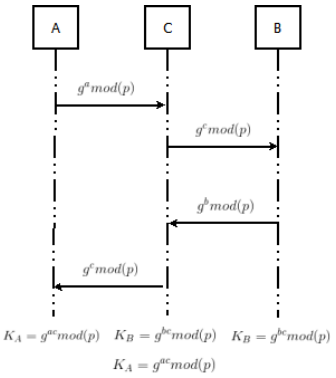

FIGURE A.1: Ataque Man in the Middle

Existen numerosas formas de prevenir los ataques Man in the Middle. Una posibilidad es lo que se llaman las firmas digitales, cuyo objetivo es asegurar la autenticación en un protocolo. Por ejemplo, podríamos usar como firma digital el DSA [28], que es un estándar del Gobierno Federal de los Estados Unidos.

Bibliografía

[1] Douglas R. Stinson. Cryptography: Theory and Practice 3rd Edition.

[2] Zeph Grunschlag. Perfect Secrecy. Columbia University. URL http://www.cs.columbia.edu/~zeph/4261/lectures/perfect-secrecy.pdf.

[3] Online Course. The Pennsylvania State University. Canonical Correlation Analysis. URL https://onlinecourses.science.psu.edu/stat505/node/65.

[4] C.K. Bhatia. Canonical Correlation Analysis. I.A.S.R.I., Library Avenue, New Delhi-110 012

[5] D. Clark. Understanding Canonical Correlation Analysis 1975. Lanchester Polytechnic.

[6] Harold Hotelling. Relations Between Two Sets of Variates. Columbia University. URL http://links.jstor.org/sici=0006-3444%28193612%2928%3A3%2F4%3C321%3ARBTSOV%3E2.0.CO%3B2-F.

[7] Magnus Borga. Canonical Correlation a Tutorial January 12, 2001.

[8] David R. Hardoon , Sandor Szedmak and John Shawe-Taylor. Canonical correlation analysis; An overview with application to learning methods.

[9] Boaz Barak. Computational Indistinguishability, Pseudorandom Generators September 27, 2007.

[10] Ayan Mahalanobis. Diffie Hellman Key Exchange Protocol, Its Generalization and Nilpotent Groups. August 2005. Florida Atlantic University.

[11] Stefan Brands. An Efficient Off-line Electronic Cash System Based On The Representation Problem. Technical Report CS-R9323, CWI. March 1993.

[12] Michael Steiner, Gene Tsudik, Michael Waidner. IMB Zürich Research Laboratory. Switzerland.

[13] Pro-technix.com The only unbreakable cryptosystem known—the Vernam cipher. Pro-technix.com. Retrieved 2014-03-17.

[14] Claude Shannon. Communication Theory of Secrecy Systems. 1949. *Bell System Technical Journal.*

[15] Bellovin, Steven M. 2011. Frank Miller: Inventor of the One-Time Pad.

[16] Taher ElGamal. 1985. A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory.*

[17] M. Abdalla, M. Bellare, P. Rogaway. DHAES, An encryption scheme based on the Diffie–Hellman Problem.

[18] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. Chapter 8.4 ElGamal public-key encryption.

[19] Dan Boneh. The Decision Diffie–Hellman Problem. *Lecture Notes in Computer Science* 1423: 48–63.

[20] Weisstein, Eric W. Correlation Coefficient. MathWorld. Wolfram Research.

[21] Härdle, Wolfgang; Simar, Léopold. 2007 Canonical correlation Analysis. *Applied Multivariate Statistical Analysis.* pp. 321–330.

[22] Knapp, T. R. 1978. Canonical Correlation Analysis: A general parametric significance-testing system. emphPsychological Bulletin.

[23] Armstrong, J. Scott. 2012. Illusions in Regression Analysis. *International Journal of Forecasting.*

[24] Yuki Arase, Xing Xie, Takahiro Hara, Shojiro Nishio. Mining people's trips from large scale geo-tagged photos. ACM Multimedia 2010.

[25] Simone Santini. Very low-rate video processing, Proc. SPIE 4311, Internet Imaging II, 279 (December 27, 2000)

[26] Boyd, J.E.; Meloche, J. ; Vardi, Y, Statistical tracking in video traffic surveillance, The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999.

[27] W. Zhao Sarnoff, R. Chellappa, P. J. Phillips, A. Rosenfeld Face recognition: A literature survey, ACM Computing Surveys (CSUR) Surveys Homepage archive, Volume 35 Issue 4, December 2003, Pages 399-458.

[28] FIPS-186, Federal Information Processing Standards Publication. Information Technology Laboratory National Institute of Standards and Technology Gaithersburg, MD 20899-8900.

[29] Benedikt Schmidt, Ralf Sasse, Cas Cremers, David Basin. Automated Verification of Group Key Agreement Protocols. IEEE Symposium on Security and Privacy (Oakland), San Jose, CA, May 2014.

[30] Rubina, Frank (1996). One-Time Pad cryptography. Cryptologia 20 (4): 359–364. doi:10.1080/0161-119691885040. ISSN 0161-1194.

[31] Fostera, Caxton C. (1997). Drawbacks of the One-time Pad. Cryptologia 21 (4): 350–352. doi:10.1080/0161-119791885986. ISSN 0161-1194.